Network of Excellence on
Engineering Secure Future Internet
Software Services and Systems

Network of Excellence

Proceedings of the first

ESSoS Doctoral Symposium

(ESSoS-DS 2012)

Eindhoven, The Netherlands

February 15, 2012

European Commission
Information Society and Media

NESSoS - 256980

Preface

This volume constitutes the proceedings of the First Doctoral Symposium on Engineering Secure Software and Systems (ESSoS-DS), held on February 15th, 2012 in Eindhoven, The Netherlands and hosted by the ESSoS 2012 Symposium. The ESSoS symposia series is one of the few conference-level events dedicated to secure software engineering. The Doctoral Symposium was organized with the help of the European Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS).

The domain of engineering secure software and systems covers a collection of engineering activities that aim at the creation of software services and systems, that are both behaviorally correct (typically guided by software engineering principles) as well as secure (typically guided by security engineering principles). In brief, software and security engineering research is combined and consolidated in an integrated and interdisciplinary engineering process with the aim of creating trustworthy software.

The challenges of the engineering process are to address a diversity of aspects such as requirements modeling and analysis of secure systems, the specification of architectural interfaces, high-level and detailed design, implementation (through the reuse and composition of existing components and services, as well as through the programming of new ones), and the validation, verification, model checking and testing of software in order to provide assurance on security aspects.

The goal of the ESSoS-DS is to provide PhD students an opportunity to discuss their research in the area of engineering secure software and systems in an international forum, and receive useful feedback from a panel of well-known experts in the field. It is an excellent chance for meeting and sharing experiences with other PhD students addressing similar topics or at a similar stage in their doctoral work. The students obtain guidance both on the academic content of their current work and regarding the potential future research trajectories.

In response to the call for papers, a total of eighteen submissions were received, which were peer reviewed by at least two referees. Of these submissions, twelve papers were accepted for presentation at the doctoral symposium and publication in this volume. The volume is available electronically through CEUR-WS.org and indexed by DBLP.

Acknowledgements

We would like to thank the ESSoS 2012 general and local organization for giving us the opportunity to organize this doctoral symposium as well as the NESSoS EU Network of Excellence for motivating us to run it. Many thanks to all authors who submitted papers to the ESSoS-DS, and particularly to the young researchers willing to discuss their ongoing work. We congratulate those whose proposal was accepted, and we hope that the comments of the reviewers have been constructive and encouraging for them and for the other authors. Our gratitude also goes to the reviewers and the members of the Program Committee, for

their timely and accurate reviews and for their feedback to the authors for improving the papers, and, even more importantly, for providing the PhD students valuable feedback for their current work and their future research plans. We also like to thank the experts who chaired the sessions. Our special thanks goes to ESSoS symposium and the NESSoS project that provided financial support for the symposium.

Munich, February 2012

Jorge Cuellar and Nora Koch
ESSoS-DS 2012 Organizers and PC Chairs

Programme Committee

Benoit Baudry	INRIA, Rennes
Manuel Clavel	IMDEA, Madrid
Jorge Cuellar	Siemens, Munich
Maritta Heisel	Universität Duisburg-Essen
Valerie Issarny	INRIA Paris-Rocquencourt
Wouter Joosen	Katholieke Universiteit Leuven (KUL)
Nora Koch	Ludwig-Maximilians-Universität München (LMU)
Javier Lopez	Universidad de Málaga
Fabio Martinelli	Consiglio Nazionale delle Ricerche (CNR), Pisa
Aljosa Pasic	AToS ORIGIN, Madrid
Christoph Sprenger	ETH Zurich
Ketil Stølen	STIFTELSEN SINTEF, Oslo
Martin Wirsing	Ludwig-Maximilians-Universität München (LMU)

Subreviewers

Artsiom Yautsiukhin
Marinella Petrocchi
Carmen Fernández-Gago
Daniele Sgandurra
Thanh Binh Nguyen
Yan Li
Gencer Erdogan
Francisco Moyano
Bjørnar Solhaug

Table of Contents

Risk-driven Security Testing versus Test-driven Security Risk Analysis *

Gencer Erdogan[1,2]
Supervised by: Ketil Stølen[1,2]

[1] Department of Informatics, University of Oslo, PO Box 1080 Blindern, N-0316 Oslo, Norway
[2] Department for Networked Systems and Services, SINTEF ICT, PO Box 124 Blindern, N-0314 Oslo, Norway
{gencer.erdogan,ketil.stolen}@sintef.no

Abstract. It is important to clearly distinguish the combinations of security testing and security risk analysis depending on whether it is viewed from a security testing perspective or a security risk analysis perspective. The main focus in the former view is security testing in which test objectives are to be achieved, while the main focus in the latter view is security risk analysis with the aim to fulfill risk acceptance criteria. The literature's lack of addressing this distinction is accompanied with the lack of addressing two immediate problems within this context, namely the gap between high-level security risk analysis models and low-level security test cases, and the consideration of investable effort. We present initial ideas for methods that address these problems followed by an industrial case study evaluation in which we have gathered interesting results.

Keywords: Security risk analysis, Security testing

1 Introduction

Security testing is a process to determine that an information system protects data and maintains functionality as intended [1]. Security risk analysis is a specialized risk analysis approach in which information security risk is associated with the potential that threats will exploit vulnerabilities of an information asset, or group of information assets, and thereby cause harm to an organization [2].

The literature collectively refers to the combinations of security testing and security risk analysis as risk-based security testing. It is, however, important to make a clear distinction between such combinations depending on whether it is viewed from (A) a security testing perspective, or (B) a security risk analysis perspective. In (A) the main focus is security testing in which test objectives are to be achieved while risk analysis is used as a means to make security testing

* This work has been conducted as a part of the DIAMONDS (201579/S10) project funded by the Research Council of Norway, as well as a part of the NESSoS network of excellence funded by the European Commission within the 7th Framework Programme.

more effective. We name this approach Risk-driven Security Testing (RST). In (B) the main focus is security risk analysis with the aim to fulfill risk acceptance criteria while security testing is used as a means to develop and/or validate risk models. We name this approach Test-driven Security Risk Analysis (TSR). Furthermore, we address two key challenges within both RST and TSR. The first challenge is to bridge the gap between high-level security risk models and low-level security test cases, while the second challenge is to make sure that the investable effort is correctly reflected in RST and TSR. As an initial evaluation of our proposed methods for RST and TSR, we have carried out an industrial case study evaluation of a TSR based approach in which we have gathered interesting results.

The paper is structured as follows: Sect. 2 outlines the research objectives followed by a brief description of the research method, Sect. 3 gives an overview of the work done to date, Sect. 4 presents preliminary results from an industrial case study in which a TSR approach was tried out, and Sect. 5 concludes the paper.

2 Research Objectives and Research Method

Security risk analysis models are often at a high-level of abstraction (e.g. business level), while security test cases are at a low-level of abstraction (e.g. implementation level), and the challenge in RST is to identify the most important security test cases, while the challenge in TSR is to identify an accurate security risk model of the target of evaluation.

These challenges are important to address in order to define exactly what to test, produce only the necessary security test cases and to obtain an accurate security risk model. The two first points provide the security testers with an indication for when to terminate the security testing process, i.e. to limit the scope of the security testing process. The necessity to limit the scope of the security testing process is due to the fact that security is often constrained by cost and time – one example from the industry is the author's personal experience of conducting security testing in a European organization [3]. It is therefore essential to take the effort available for conducting an RST or a TSR into account.

The author seeks to address the particular task of developing an industrial guideline for effort dependent risk-driven security testing and test-driven security risk analysis. The initial research question is the following:

– What is a good industrial guideline for effort dependent risk-driven security testing and test-driven security risk analysis?

The research work will adopt an iterative incremental approach where the industrial partners provide industrial case studies on RST and TSR. There will, in total, be carried out six case studies. In this light, the research process will be conducted using the Technology Research Method [4] which is an iterative incremental research method.

3 Current Work

Figure 1 presents the steps in RST and TSR. The steps for security testing are in line with the steps presented in the Standard for Software Component Testing [6], which is also one of the building blocks in the upcoming new international standard for software testing - ISO/IEC 29119 [7]. The steps for security risk analysis are in line with the steps presented in ISO 31000 [8].

In **RST** security testing is supported by security risk assessment in order to make security testing more effective. The aim is to focus the security testing process to carry out security tests on the most important parts of the system under test, and to execute only the most important security test cases. RST provides two alternatives to achieve this aim. **The first alternative** (Step 3) achieves this by first directing the identified security test model from Step 2 as input to Step 3. Based on this input, risks concerning the system under test are identified, estimated and evaluated in Step 3. The output of Step 3 is a risk evaluation matrix that contains a list of prioritized risks of the system under test. The risk evaluation matrix is finally used as a basis for generating and prioritizing security test cases in Step 4. **The second alternative** (Step 5) achieves this by first directing the identified security test cases from Step 4 as input to Step 5. Based on this input, risks addressed by the security test cases are identified, estimated and evaluated in Step 5. The output of Step 5 is a risk evaluation matrix that contains a list of prioritized risks of the system under test. The risk evaluation matrix is finally used as a basis for eliciting the most important security test cases to be executed in Step 6. The dashed rectangles that surrounds Step 3 and 5 indicate that the security testing process can be supported by either Step 3 *or* Step 5 when conducting the RST method.

In **TSR** security risk analysis is supported by security testing in order to develop and/or validate risk models. The aim is to strengthen the correctness of the security risk analysis models. TSR provides two alternatives to achieve this aim. **The first alternative** (Step 2 and 3) supports the *development* of risk models by identifying potential risks. This is achieved by first directing the system model and the identified assets of the target of evaluation from Step 1 as input to Step 2. Based on this input, security test cases are generated and prioritized in Step 2. The identified security test cases are then executed in Step 3. Finally, the security testing results are used as a basis for identifying potential risks which are used as an input to Step 4. **The second alternative** (Step 5 and 6) supports the *validation* of risk models by first identifying security tests that explore the risks and then validating the risk model based on the security testing results. This is achieved by first directing the risk evaluation matrix from Step 4 as input to Step 5. Based on this input, security test cases are generated and prioritized in Step 5. The identified security test cases are then executed in Step 6. Finally, the security testing results are used as a basis for validating and updating the security risk analysis models. The dashed rectangles that surrounds Step 2 and 3 and Step 5 and 6 indicate that the security risk analysis process can be supported by security testing for either developing risk models or validating risk models, *or both* when conducting the TSR method.

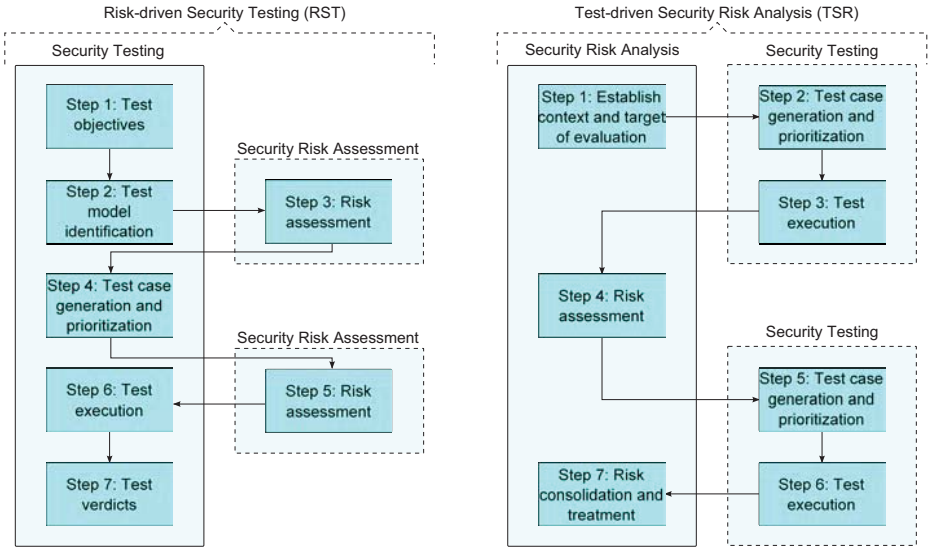

Fig. 1. *The steps in Risk-driven Security Testing and Test-driven Security Risk Analysis.*

4 Case Study

We have carried out an industrial case study using a TSR based approach. In particular, we carried out the alternative for *validating* the risk model, i.e. step 1, 4, 5, 6 and 7 of the TSR approach, as explained in Sect. 3. The target system analyzed was a multilingual, web-based e-business application, which serves as the backbone for the system owner's business goals and is used by a large amount of users every day. The case study was carried out in a period of four months in form of meetings and security testing sessions. The presented results are limited to the experiences obtained in the case study due to confidentiality reasons.

Table 1 outlines the process undergone during the case study. The first column specifies the meeting sequence (SRA denotes a security risk analysis meeting, while ST denotes a security testing meeting). The second column lists the participants (C denotes participants from the customer organization, while A denotes participants from the analysis team). The third column describes the contents and achievements in each meeting. Finally, the fourth column shows the approximate time spent (in man-hours) for each meeting. The time spent on work before and after the meetings is not included in the table.

Security risk analysis was conducted using the CORAS approach [5]. The process of identifying security tests was carried out by first using the risk evaluation matrix with identified risks as a starting point. Then, the threat scenarios that lead up to a risk that needed to be treated were systematically identified as *testable* or *not testable*. Furthermore, the identified testable threat scenarios were prioritized based on their individual effort for realizing the risk. Finally, secu-

Table 1. The process undergone during the case study

Meeting	Participants	Contents	Hours
1 - SRA	**C**:One domain expert. **A**:The analyst. Two domain experts.	Defining the goals, context, target, focus and scope of the SRA.	2
2 - SRA	**C**:One domain expert. One developer. **A**:The analyst. The secretary. One domain expert.	Defining the goals, context, target, focus and scope of the SRA.	3
3 - SRA	**C**:One domain expert. **A**:The analyst. The secretary.	Concretizing the scope, assets and risk evaluation criteria of the SRA.	6
4 - SRA	**C**:One domain expert. **A**:The analyst. The secretary.	Identifying and evaluating risks.	6
5 - SRA / ST	**C**:One domain expert. **A**:The analyst. The secretary.	Identifying security tests.	6
6 - ST	**A**:The analyst. The secretary.	Implementing security tests.	8
7 - ST	**A**:The analyst. The secretary.	Executing security tests.	6
8 - ST	**A**:The analyst. The secretary.	Executing security tests.	6
9 - SRA	**C**:One domain expert. One developer. **A**:The analyst. The secretary.	Validating and updating the risk model based on the security testing results. Suggesting treatments for security tests that failed.	2

rity tests were identified for the prioritized testable threat scenarios. A testable threat scenario, in this context, is simply a threat scenario that is possible to test at the software level.

Table 2 presents the overall TSR results obtained in the case study. The leftmost column specifies the information security assets taken into consideration during the security risk analysis process. The row named *Common* denotes the number of security risk analysis elements that are common for all assets. The topmost row specifies the number of security risk analysis elements: R denotes the number of risks, TS denotes the number of threat scenarios, RT denotes the number of tested risks, TST denotes the number of tested threat scenarios (the tested threat scenarios that addressed confidentiality also addressed integrity and are therefore not counted two times in the total sum), R *Upd.* denotes the number of updated risks based on the security testing results and TS *Upd.* denotes the number of updated threat scenarios based on the security testing results.

A total number of 31 risks and 43 threat scenarios were identified during the risk assessment, and from these, 11 risks and 7 threat scenarios were tested. Approximately 80% of the identified security tests that explored these risks and threat scenarios uncovered some form of security vulnerability. This is an indication that the risk model contributed to identify relevant security tests. Furthermore, the security testing results helped us to validate and update the risk model; approximately 20% of all risks and 14% of all threat scenarios had

Table 2. Test-driven Security Risk Analysis results

SRA elements / Assets	R	TS	RT	TST	R Upd.	TS Upd.
Confidentiality of information	8	2	5	5*	3	0
Integrity of information	8	7	5	5*	3	1
Availability of information	11	13	1	2	0	0
Accountability of information	4	6	0	0	0	0
Common	0	15	0	0	0	5
Total	31	43	11	7	6	6

to be adjusted, with respect to likelihood values, based on the security testing results.

5 Conclusion

The combinations of security risk analysis and security testing must be clearly distinguished depending on whether it is viewed from a security testing perspective (RST), or a security risk analysis perspective (TSR). Additionally, the immediate challenges that need to be addressed in these approaches are: (1) the gap between high-level security risk analysis results and low-level security test cases, and (2) the correct reflection of investable effort. An industrial case study evaluation of a TSR based approach showed how security testing can be used as a significant means to validate and update the risk model – i.e. approximately 20% of all risks and 14% of all threat scenarios had to be adjusted, with respect to likelihood values, based on the security testing results.

References

1. The Committee on National Security Systems: National Information Assurance (IA) Glossary, CNSS Instruction No. 4009, 2010.
2. ISO/IEC 27005:2011(E): Information technology - Security techniques - Information security risk management.
3. Erdogan, G., Meland, P.H., Mathieson, D.: Security Testing in Agile Web Application Development - A Case Study Using the EAST Methodology. In Proceedings of XP'2010. pp.14–27, 2010.
4. Solheim, I., Stølen. K.: Technology research explained. SINTEF Report, A313. Technical report, SINTEF Information and Communication Technology, 2007.
5. Lund, M.S., Solhaug, B., Stølen, K.: Model-Driven Risk Analysis: The CORAS Approach. Springer, 2011.
6. British Computer Society Specialist Interest Group in Software Testing: BS 7925-2 Software testing. Software component testing, 1998.
7. Working Group 26 (WG26) of the ISO/IEC JTC1/SC7 Software and Systems Engineering committee. `http://www.softwaretestingstandard.org/` Last date accessed 2012-02-07.
8. International Organization for Standardization: ISO 31000 Risk management - Principles and guidelines, 2009.

Security of the OSGi platform *

Anton Philippov, Olga Gadyatskaya, and Fabio Massacci

DISI, University of Trento, Italy
{name.surname} @unitn.it

Abstract. In the last few years we have seen how increasing computational power of electronic devices triggers the functionality growth of the software that runs on them. The natural consequence is that modern software is no longer single-pieced, it becomes, instead, the composition of autonomous components that run on the shared platform. The examples of such platforms are web browsers (such as Google Chrome), smartphone and smart card operating systems (e.g., Android and Java Card), intelligent vehicle systems or smart homes (usually implemented on OSGi). On one hand, these platforms protect components by isolation, but at the same time, provide methods to share and exchange services. If the components can come from different stakeholders, how do we make sure that one's services would only be invoked by one's authorized siblings? In this PhD proposal we illustrate the problems on the example of OSGi platform. We propose to use the security-by-contract methodology (S×C) for loading time security verification to separate the security from the business logic while controlling access to applications.

1 Introduction

With the ever-increasing popularity of smart devices and rapid development of Internet, the single application model is more and more often being replaced with the service platforms. These include Java Card platform for multi-applicational smart cards, Android for smartphones, OSGi for component-based Java applications and smart homes, Google Chrome platform for browser plugins. In general, we say that *service platform* is a platform, on which applications are isolated, but can share selected parts of their functionality with other applications on the platform. Furthermore, such shared functionality we will call a *service*. In the current proposal we will concentrate on Java-based service platforms and, in particular, OSGi platform.

The Open Services Gateway Initiative (OSGi) framework [1] is one of the most flexible solutions for the deployment of pervasive services in home, office, or automobile environments [5, 7, 9]. The OSGi services are also the basic building blocks for service mash-ups extending the classical "smart homes" scenarios to richer settings [8]. In a nutshell, the OSGi framework redefines the modular system of Java by introducing *bundles*: JAR files enhanced with specific

* This work is partially supported by EU-funded project FP7-257930 ANIKETOS and EU-funded project FP7- 256980 NESSOS.

metadata. The *services* layer connects bundles in a dynamic way with a publish-find-bind model for Java objects. As a result, an OSGi platform is expected to be highly dynamic. All pervasive and mash-up applications expect that bundles can be installed, updated or removed at any time. From a security perspective, the possibility of bundle interactions is a threat for bundle owners. Since bundles can contain sensitive data or activate sensitive operations (such as locking doors and windows of somebody's house), it is important to ensure that the security policy of each bundle owner is respected by other bundles. However, such aspects have been only partially investigated.

How do we make sure that one's services are invoked by one's authorized siblings? A simple solution is to rely on service-to-service authentication to identify the services and then interleave functional and security logic into bundles, for example, by using aspect-oriented programming [9]. However, this decreases the benefits of common platform for service deployment and significantly hinders evolution and dynamicity: any change to the security policy would require redeployment of the bundle (even if its functionalities are unchanged). Vice versa, any changes in the bundle's code would require redeployment of security as well.

Our solution is to use the security-by-contract methodology (S×C) [2, 3] for loading time security verification in order to separate security and the business logic while achieving a sufficient protection of applications among themselves.

In the next section we illustrate the problem by introducing a concrete case study for home gateways (§2) and discuss the security issues that the plain OSGi model cannot solve without ad-hoc security codes within each bundle. We then introduce the solution (§3) and conclude in §4 with an overview of the paper.

2 Problem statement

Further we concentrate on OSGi platform, identify the flows of its security mechanisms. However, due to the similarities in the architecture, the problems can be applied to other service platforms (e.g., Android) as well. Due to the paper length constraints we skip the technical description of the OSGi platform and assume the reader has at least a basic familiarity with the service platform architecture.

The Scenario We consider as a case study an OSGi platform deployed as a service gateway in a smart home. Let us consider Alice, the smart home resident, and a telecom provider, the owner of the platform. Alice can download bundles for entertainment (news RSS feeds, media bundles from TV providers) or even bundles with traditional Internet content (like Facebook or Twitter), as nowadays new TV sets can be used for all these purposes. The interested reader can refer to [6] for more details on the news feed scenario. In this example we have used fictional names, but they give an idea of realistic bundle interactions and possible policies regarding these interactions.

Alice, a beginner stock market player, downloads and installs bundle A from provider $FSM.com$ that can provide her with an interface of the stock market

operations. This bundle includes service S_A that retrieves updates about the stock prices. However, Alice later finds and installs another stock market bundle B from $BH.fr$ provider, that also provides service for prices information retrieval S_B and service S_{fr} that allows Alice to transfer money from her stock market account (registered on $BH.fr$) to her Happy Farm account on $FB.com$. Thus Alice also installs Happy Farm bundle F.

The bundle providers want to ensure that their security policies related to bundles and services usage are enforced on the Alice's platform. Their requirements are as follows:
FSM.com: *Access to S_A service is allowed only for bundles signed by $FSM.com$.*
BH.fr: *Access to S_B service is allowed only for bundles signed by $BH.fr$. Only bundles signed by $BH.fr$ can import the package containing S_B. Access to S_{fr} service can be granted only for bundles signed by $FB.com$ or by $BH.fr$.*

The OSGi platform at Alice's smart home has to ensure that the requirements of each provider are respected. We will next discuss how the OSGi platform can enforce these requirements and why this approach is not satisfactory. We will also demonstrate that there can exist similar requirements of bundle providers that cannot be enforced by the OSGi platform at all.

Security Challenges Let us first briefly present the relevant OSGi platform details [1]. An OSGi bundle is a JAR file that includes the manifest.mf file containing the necessary OSGi metadata including dependencies and the provided libraries. Dependencies are expressed as *requirements* on *capabilities*. Capabilities are attribute sets in a specific namespace and requirements are filter expressions that assert the attributes of the capabilities. A requirement is satisfied when there is at least one capability that matches the filter. Bundles can interact through two complementary mechanisms: the export/import of packages and the service registration/lookup facility. A *service* is a normal Java object registered under a Java interface with the *service registry*. Each bundle is associated with a set of permissions, that are queried at runtime. The OSGi specification defines ServicePermission, BundlePermission and PackagePermission, which are used for getting/registering a service, importing/exporting bundles and packages respectively. The platform can authenticate code by download location or by signer (digital signature). The Conditional Permission Admin service manages the permissions based on a comprehensive conditional model.

We assume the framework can host multiple third-party bundles, and these bundles can freely register services. The goal of the telecom provider running the platform is to make sure that there are no undesired security or functionality problems among different bundles installed by the end user (who most likely does not even know what is a bundle and just sees the web interfaces of the services). Thus a threat scenario under investigation is a case when a bundle gains unauthorized access to the sensitive data of another bundle (security threat), or a bundle is malfunctioning due to unavailability of a certain service (functionality threat). We now discuss these threats separately in the light our scenario.

A confidentiality attack can be realized by the bundle A of provider $FSM.com$ getting access to the sensitive stock market prices service S_B of provider $BH.fr$.

This might happen if A imports the package containing the service S_B definition, requires the bundle B (thus importing all its exported packages), or tries to get a reference to this service from the Service Registry and then get access to the object referenced.

We now discuss how the current OSGi security management can address this security threat. Import of a package or a require-bundle action can be granted if the requiring bundle has corresponding permissions. Simple reviewing of the manifest file and permissions file of the bundle A can report about a (potential) attempt to interact with the bundle B. However, there is no convenient and simple way for the owner of the bundle B, the $BH.fr$ provider, to declare which other bundles are allowed to import its packages.

Package importing can be guarded by the permissions mechanism, as we discussed before. Currently only the platform owner (the telecom provider) can define and manage policies in the Conditional Permission Admin policy file. The $BH.fr$ provider might contact the telecom provider to ask him to set the required permissions, or its bundle B, being granted the necessary permissions, can add new permissions to the Conditional Permission Admin policy file. These approaches are organizationally cumbersome and costly, as they require the operator to push the changes to its customers before any downloads of $BH.fr$ bundles, even the customers have no intention of using them.

Service usage is another, more trickier issue. Again, the necessary authorizations for the service usage (more precisely, GET permissions for service retrieval) can be delivered within bundle contracts and incorporated into the policy file of the system. But the invocations of the methods within a service, once the necessary reference is obtained, are not guarded by the permission check, and usually the security checks are placed directly within the service code, thus mixing the security logic with the execution logic.

Another solution, that is traditional for mobile Java-based component systems, could be to ask Alice each time a specific permission is needed. But Alice is not the owner of the bundles to make such decisions, nor is she interested to do so. Let us consider a more complex scenario now.

Example 1 *Alice wants to install the Sims add-on from the EA.com provider. This add-on is packaged into the bundle C and it will provide an integration of the Happy Farm account with her the Sims account. The functional requirement of the EA.com provider is the following: "The bundle C can be installed if and only if the F bundle is available on the platform and provides the Happy Farm service S_F.*

The requirement in Example 1 means that bundle C can be installed only if the service S_F is already provided on the platform. This requirement prevents the denial of service by the Sims bundle. The bundles are running on top of a single JVM, thus the denial of service attack can cause a restart of the whole system [4]. This functional requirement is, in fact, unsupported by the current OSGi specification. Requirements/capabilities model cannot provide guarantees on the provided services (except that their definition exists on the platform).

3 Proposed solution

Our proposed solution is to adopt the Security-by-Contract that was initially investigated and implemented by Bielova et al. for mobile Java-based devices [2] and by Dragoni et al. for the Java Card platforms [3]. Further we provide details on possible architecture of S×C for OSGi.

The S×C framework consists of two main components: the ClaimChecker and the PolicyChecker. The verification workflow is described on Figure 1.

Informally, the S×C process starts when a new bundle B is loaded. The ClaimChecker component then accesses the manifest file, retrieves the information about imported and exported packages and obtains the bundle contract. Then the ClaimChecker reads the permissions.perm file, which contains local bundle permissions, extracts permissions requested by the bundle B and related to services retrieval, packages importing, requirements of bundles, etc., and combines this information into the overall "security claims and needs" of the bundle. Having these claims, the ClaimChecker then analyzes the bytecode of the bundle to verify that the claims match actual code. If the verification fails, meaning that the claims are not supported by the code, the bundle is removed from the platform. Otherwise, the PolicyChecker component receives the result from the ClaimChecker and matches it with the security policy of the platform, that aggregates the security policies of all the installed bundles, and with the functional state of the platform (installed bundles, running services, etc.). If the PolicyChecker failed on either of the checks, the bundle is removed from the platform. Otherwise, it is installed and the security policy of the platform is updated by including the security requirements of B.

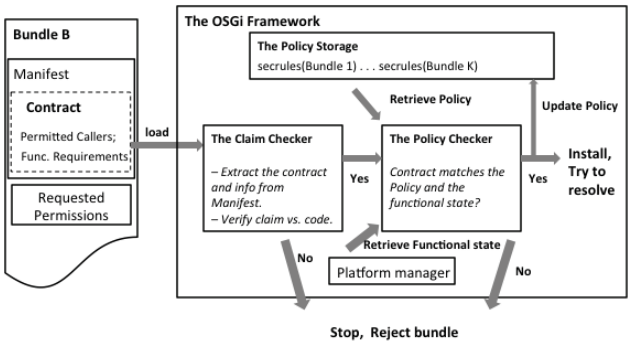

Fig. 1. SxC Workflow

In terms of technical realization, the S×C steps can be integrated with the OSGi framework. The key requirement is getting the correct and up-to date information about the state of the platform and being able to access the received bundle *before* it is deployed on the platform. The S×C framework itself can be

a bundle, provided it can access the service registry, the framework policy file, the lifecycle layer and the manifests of the bundles.

4 Conclusions

In this proposal we have identified the security problems of some of the Java-based service platforms on the example of OSGi platform. We have presented and idea of solution, which is a Security-by-Contract paradigm for the OSGi platform. We discussed the security and functionality challenges and proposed how to enable the bundle providers with ability to effectively express their security and functional requirements on the platform.

The main benefits that the S×C approach can bring to service platforms are the following. From the security aspect, the bundle providers can now specify the authorizations for access to their bundles, packages and services. The policies can be updated easily and the update does not require an interaction from the platform owner, an access to the framework policy file or an update of the execution logic of the bundle. For the functionality aspect, the bundle providers have now a more powerful tool for expressing their functional requirements than the requirement/capability model of OSGi. The contracts can express requirements on the current state of the platform (including requirements on the states of the bundles or certain services provision, or absence of the competitor's resources).

References

1. T. O. Alliance. OSGi service platform core specification. Version 4.3, 2011.
2. N. Bielova, N. Dragoni, F. Massacci, K. Naliuka, and I. Siahaan. Matching in security-by-contract for mobile code. *Journal of Logic and Algebraic Programming*, 78(5):340 – 358, 2009.
3. N. Dragoni, O. Gadyatskaya, and F. Massacci. Can we support applications evolution in multi-application smart cards by Security-by-Contract? In *WISTP-2010*, LNCS 6033, pages 221–228.
4. N. Geoffray, G. Thomas, G. Muller, P. Parrend, S. Frénot, and B. Folliot. I-JVM: a Java Virtual Machine for Component Isolation in OSGi. In *DSN'2009*. IEEE.
5. T. Gu, H. Pung, and D. Zhang. Toward an OSGi-based infrastructure for context-aware applications. *IEEE Perv. Computing*, 3:66–74, 2004.
6. F. Innerhofer-Oberperfler, S. Löw, R. Breu, M. Breu, M. Hafner, B. Agreiter, M. Felderer, P. Kalb, R. Scandariato, and B. Solhaug. D2.2: A configuration management process for lifelong adaptable systems. Public deliverable of the Secure Change project, 2011.
7. C. Lee, D. Nordstedt, and S. Helal. Enabling smart spaces with OSGi. *IEEE Perv. Computing*, 2(3):89 – 94, 2003.
8. A. Ngu, M. Carlson, Q. Sheng, and H. Paik. Semantic-based mashup of composite applications. *IEEE Tran. on Services Computing*, 99:2–15, 2010.
9. P. Phung and D. Sands. Security policy enforcement in the OSGi framework using aspect-oriented programming. In *COMPSAC'2008*, pages 1076–1082.

Security Guarantees and Evolution: From Models to Reality*

Martín Ochoa[1,2]

[1] Siemens AG, Germany
[2] TU Dortmund, Germany
martin.ochoa@cs.tu-dortmund.de

Abstract. Achieving security in practical systems is a hard task. As it is the case for other critical system properties (i.e. safety), security should be a concern through all the phases of software development, starting with the very early phases of requirements and design. Although the arguments in favour of formal verification at the design level are many (rigour and cost-benefit being at the top of the list), answers to two relevant questions about the limitations of this approach are crucial for its success and further application in industrial contexts: Is security verification at the design phase scalable under continuous evolution? What are the limits of the formal security guarantees achieved at the model level when software is ultimately deployed? In this paper we report on recent results and work in progress towards a better understanding of these two fundamental questions.

1 Motivation

In recent years, information security has gained increasing attention from the general public and there is a consensus about its paramount importance in society. Examples include recent scandals on users private data [11], leaks of government secret documents and public threats from anonymous hacker groups to corporate and governmental IT systems worldwide [1,2]. Long gone are the days where the term 'computer security' was associated exclusively with spies, conspirational theories and cryptography. Today most successful attacks exploit vulnerabilities related to problems in design or implementation rather than vulnerabilities in cryptographic mechanisms. And most of the attackers are motivated teenagers, typically not interested in the mathematical aspects of cryptography.[3]

There are several reasons why security is difficult to achieve in practice. On the one hand, the complexity of modern system architectures is constantly increasing: software logic evolves, often driven by the market pressure to deliver new functionalities, and different operating systems and hardware configurations

* PhD work supervised by Jan Jürjens at the Chair for Software Engineering of the TU Dortmund and Jorge Cuéllar at Siemens Corporate Research.

[3] Also social aspects of security result in attacks in many cases, but those are very difficult to control by technological means and are out of the scope of this work.

make each instantiation of an IT system unique. Moreover, software components from different producers delivered without accurate (if at all) security guarantees are used together to achieve customized solutions.

Techniques to tackle this ever 'moving target' exist in different areas of computer science and engineering: from software and hardware formal verification to testing, but also at the level of business-processes modelling and risk-analysis. In fact, security plays a role at different levels of abstraction and at different phases of the development cycle, and if one wants to have a high degree of assurance about the security of a system one should consider them all.

In general, the strongest guarantees about software and hardware behaviour are delivered by formal methods, due to their rigour and precision. It is thus natural to consider formal methods to validate a system with respect to well-defined, mathematical descriptions of security. Nevertheless, formal methods are difficult to apply in realistic software-development scenarios because they require highly specialized designers and programmers in order to carry out the formalizations and because many verification tools available hardly scale to realistic scenarios, in particular at the level of implementations. Additionally, the security requirements of a system are not always precise, because often they are formulated in natural language.

At the present time, the application of formal methods seems to be more reasonable to take place at the level of system design, where models are abstracted from implementation details and are more amenable to automated verification. Since some security problems can be detected already at the specification level, the cost-benefit of applying this methodology is typically better than repairing design problems at later stages of development. Since the de-facto standard for system modelling in industry is the Universal Modelling Language (UML), it is natural to perform this formal verification on UML models, if one aims at industry acceptance. It is however not enough to have strong security guarantees on system models to be able to judge the overall security of a deployed system, which is the ultimate goal. As argued before, it is difficult to verify the code because of its size and the huge variety of programming languages and paradigms. Moreover it can be the case that libraries or components from third parties are used whose source code is unavailable.

Unfortunately, security is a never ending open loop, since no matter how strong guarantees a given system has, new exploits appear that consider properties or interfaces that were not considered at design. This is prominently illustrated by side-channels: here the attacker exploits information such as electromagnetic radiation, timing and shared memory behaviour to gain possession of confidential data. Many of these side-channels are difficult to capture since they rely on micro-architectural configurations such as the duration of processor instructions or the cache behaviour. Closing these interfaces is sometimes impossible or costly, because of the impact to other system requirements (i.e. efficiency).

In summary, we believe that it is unrealistic to attempt to build complex industrial systems with a 100% guarantee of security because logical or physical

interfaces that are vulnerable to attacks are often only exposed after real attacks take place. To date there is also no standard single methodology, tool, or model to tackle the whole Software Development Life-Cycle. It is thus necessary to provide tools that help to cope with evolution problems at all levels of abstractions and during the whole life-cycle. It is our believe that the idea that systems are going to be secure because we commit to a single approach (for example rigorous security requirements elicitation, industrial best-practices, strict patch update policies etc.) is fallacious: we have to work on all phases and at different abstraction levels, sometimes using different models and different methodologies. In this work we propose a set of such tools, easy to use by end users, and addressing the aforementioned problems on Models, Implementations and Micro-architectures.

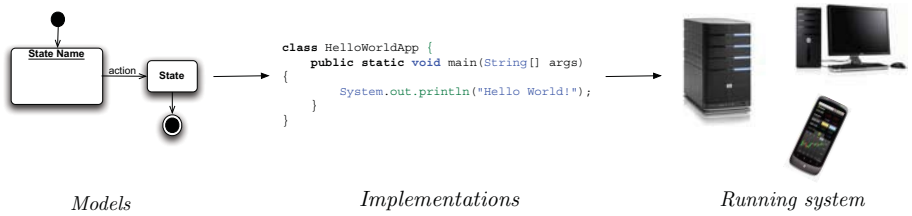

Fig. 1. Roadmap from models to running systems

2 Roadmap and contributions

In the following we summarize the goals of this research and the achieved results so far. Security is typically described as the conjunction of one or more security requirements, abstractly classified as Confidentiality, Availability and Integrity. In this work we consider mainly Confidentiality and Integrity: we want to understand how information is flowing from a group of users to another. There are basically two ways to look at the problem and our proposed solutions: a) according to the abstraction level modelled and b) according to the security properties considered. First, we will take the first point of view (see Fig. 1).

Models We consider the problem of specification evolution for security at the level of UML models. We extend the UMLsec [5] notation and verification techniques to reason about changes in the specification by means of the novel UMLseCh notation [7,6]. For some properties defined in static UML diagrams, we describe sufficient conditions that soundly preserve the security of already verified models by analysing the delta implied by the modifications. This fine-grained incremental technique is a good choice for structural of properties because of their locality: changes of parts of the model usually affect the property in a clearly identifiable, small subset of the specification. For behavioural models, incremental changes can affect security in non-trivial ways. Therefore we propose to focus on compositionality results: if one or more components of a given system evolves, the overall security of the system can be decided (efficiently) by re-verifying the evolved components exclusively given an (efficient)

compositionality condition. We describe such a decision procedure for secrecy on cryptographic protocols specified as Sequence diagrams [10] that is sound and complete with respect to a previous verification technique proposed by Jürjens [5]. We currently extend previous work on verifying non-interference in UML state-charts and derive compositionality results for interacting objects.

Implementations For evaluating the security of implementations we consider model-based testing of security requirements based on a black box model of the system. The methodology proposed is well-founded with respect to the requirements considered, and extends previous work on security testing [4]. We also discuss conditions under which the security relevant properties of the model are preserved under incremental changes [3].

Micro-architecture At this detailed abstraction level, we focus on cache configurations and how they can act as a leaking channel for different adversaries. Configurations of the CPU play a determinant role on the security of the system with respect to side-channel attacks, and the change in configurations is a typical phenomenon of system's evolution. Avoiding the use of caches conflicts with efficiency requirements and is therefore not realistic for a wide range of systems. A promising technique to achieve formal guarantees about countermeasures striving for a trade-off between security and other conflicting requirements is quantitative information flow analysis (for example [8]). We formalize heuristic countermeasures proposed in the literature and give strong security guarantees for arbitrary programs under a well-defined attacker model [9], and validate our approach using an automatic tool chain that evaluates compiled programs for various architectures.

Notice that we do not aim at a formal integration of the security guarantees obtained at the discussed levels of abstraction: such a task, although interesting, goes outside the scope of this work. On the other hand, current industrial environments do not have a formally justified software development process. We consider different layers of abstraction mainly due to necessity: any error found in any of these layers would invalidate the over-all security of the system.

It is nevertheless interesting to informally discuss our contributions from the perspective of the security properties considered and the assumptions they rely on at different abstraction levels. In fact, when analysing information-flow at the model level, we are assuming perfect mechanisms for access control, and among others, perfect cryptography, which guarantees access control when information is shared through insecure networks. To gain confidence in the correctness of those mechanisms, we validate them locally using model-based testing and performing a Dolev-Yao analysis for the cryptographic protocol logic. To gain even more confidence about primitives, we consider the micro-architectural abstraction level, using quantitative information-flow related techniques. We can see this as an (informal) chain of assumptions and guarantees across abstraction levels, as depicted in Fig. 2.

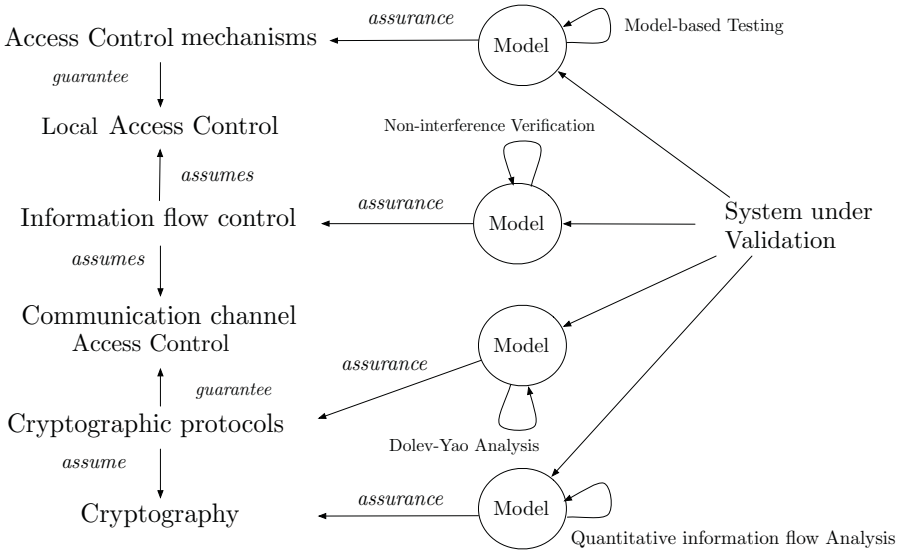

Fig. 2. Access control and Information-flow control vs. model validation

The assumptions and guarantees discussed here are by no means complete: for example we are not considering operating system access control mechanisms or semantic security properties of the cryptographic primitives. As already discussed we believe that a complete and formally justified methodology is unrealistic. It is nevertheless essential to consider different levels of abstraction and development phases to achieve a good degree of confidence in the system, since an error in any of them would invalidate the results at higher abstraction levels or previous phases. For example a faulty implementation of access control would invalidate a secure abstract design w.r.t non-interference, and a cryptographic implementation with side-channels would make a formally verified protocol against the Dolev-Yao adversary meaningless.

3 Conclusions and Work in Progress

Software security is a difficult problem because it is a moving target, and it should be addressed at different levels of abstraction and in all phases of software development. Although a promising methodology to achieve practical security is formal verification at design time, to date there a number of limitations to this approach, in particular when the system undergoes continuous evolution. We have reported on results towards a scalable security verification at the model level, and pinned down many assumptions made at different levels of abstractions, that are vital for the formal model analysis to be sound. Our focus is to devise methodologies supporting intuitive tools, aiming at minding the gap between formal methods and industrial acceptance. At this point of our research,

we have already achieved many of the original goals, and promising work in progress is being done on the missing items. Concretely, at this stage work in progress is being done for information flow analysis: at the model level, we are interested in the automatic and efficient verification of UML state-charts; at the micro-architectural level novel formal quantification techniques are being studied that provide strong security guarantees on realistic modern processor models for powerful attackers and multiple hardware configurations. Submissions to international conferences in software engineering and automatic verification in both of these subjects are on preparation at the time of writing this manuscript.

Acknowledgements This research was partially supported by the MoDelSec Project of the DFG Priority Programme 1496 "'Reliably Secure Software Systems – RS^3" and the EU project NESSoS (FP7 256890).

References

1. LulzSec hackers claim CIA website shutdown. BBC news `http://www.bbc.co.uk/news/technology-13787229`, 2011. Online, accessed 04-Feb-2012.
2. LulzSec takes down Brazil government sites. Cnet news `http://news.cnet.com/8301-1009_3-20073219-83/lulzsec-takes-down-brazil-government-sites/`, 2011. Online, accessed 04-Feb-2012.
3. E. Fourneret, F. Bouquet, M. Ochoa, J. Jürjens, and S. Wenzel. Vérification et test pour des systèmes évolutifs. In *Approches Formelles dans l'Assistance au Dveloppement de Logiciels (AFADL)*, 2012.
4. E. Fourneret, M. Ochoa, F. Bouquet, J. Botella, J. Jürjens, and P. Yousefi. Model-based security verification and testing for smart-cards. In *Proceedings of the 6th International Conference on Availability, Reliability and Security (ARES)*, pages 272–279. IEEE, 2011.
5. J. Jürjens. *Secure Systems Development with UML*. Springer, 2005.
6. J. Jürjens, L. Marchal, M. Ochoa, and H. Schmidt. Incremental Security Verification for Evolving UMLsec models. In *Proceedings of the 7th European Conference on Modelling Foundations and Applications (ECMFA)*, volume 6698 of *Lecture Notes in Computer Science*, pages 52–68. Springer, 2011.
7. J. Jürjens, M. Ochoa, H. Schmidt, L. Marchal, S. H. Houmb, and S. Islam. Modelling secure systems evolution: Abstract and concrete change specifications. In *Proceedings of the 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM)*, volume 6659 of *Lecture Notes in Computer Science*, pages 504–526. Springer, 2011.
8. B. Köpf and M. Dürmuth. A provably secure and efficient countermeasure against timing attacks. In *Proceedings of the 22nd IEEE Computer Security Foundations Symposium (CSF)*, pages 324–335. IEEE Computer Society, 2009.
9. B. Köpf, L. Mauborgne, and M. Ochoa. Automatic quantification of cache side-channels. Cryptology ePrint Archive, Report 2012/034, 2012. `http://eprint.iacr.org/`.
10. M. Ochoa, J. Jürjens, and D. Warzecha. A sound decision procedure for the compositionality of secrecy. In *Proceedings of the 4th International Symposium on Engineering Secure Software and Systems (ESSoS)*, 2012.
11. M. J. Schwartz. Sony sued over playstation network hack. Information Week, `http://www.informationweek.com/news/security/attacks/229402362`, 2011. Online, accessed 04-Feb-2012.

Attack-preserving Security Protocol Transformations

Binh Thanh Nguyen, David Basin, and Christoph Sprenger

Institute of Information Security, ETH Zurich, Switzerland
{thannguy,basin,sprenger}@inf.ethz.ch

Abstract. The rigorous incremental development of security protocols has so far received much less attention than protocol verification techniques. In this work, we study security protocol transformations. These can serve both for simplifying protocols before verification and, in the other direction, for developing protocols by stepwise refinement of simple abstract protocols into complex concrete ones. The transformations preserve attacks on a class of security properties. Our work aims to improve our understanding of modifications of existing protocols and to enable the systematic development of entire families of new protocols. This complements existing work on post-hoc protocol verification.

1 Introduction

It is well-known that security protocols are notoriously hard to get right. This motivates the use of formal methods for their design and development. In the last decade, we have witnessed substantial progress in the formal verification of security protocols. However, methods for developing security protocols have received much less attention and protocol development remains more an art than a science.

Sprenger and Basin [14,15] have recently proposed a hierarchical development method for security protocols based on stepwise refinement that spans several levels of abstraction. Each development starts from abstract models of security properties and proceeds down to cryptographic protocols secure against a Dolev-Yao intruder. The development process traverses intermediate levels of abstraction based on message-less protocols and communication channels with authenticity and confidentiality properties. Security properties, once proved for a given model, are preserved by further refinements. They have applied their method to develop families of authentication and key transport protocols. However, developers may not be familiar with these abstractions and the underlying refinement framework. They are more familiar with cryptographic messages and transforming these messages to create new protocols from existing ones.

This motivates our study of refinements in terms of protocol transformations at the level of cryptographic messages. In particular, we are interested in protocol transformations that preserve attacks against a given set of security properties from concrete protocols to abstract ones (or, equivalently, the satisfaction of such properties in the reverse direction). Such transformations can serve the systematic development of individual protocols and entire families of protocols. Moreover, they can be applied to modify or compare existing protocols and understand their differences. The modification of security protocols is particularly error-prone (see, e.g., [2]). Security protocol standards

constitute another relevant application field for protocol transformations, since they typically comprise numerous protocol variants and options.

Security protocol transformations can also be considered as abstractions (i.e., from concrete to abstract). Hui and Lowe [10] define several kinds of attack-preserving transformations with the aim of simplifying protocols so that they can be effectively verified using model checking. They define criteria for the preservation of secrecy and authentication properties, and prove for each kind of transformation that it satisfies these criteria.

Datta et al. [6,5] use protocol templates with messages containing function variables to specify and prove properties of protocol classes. Their notion of refinement is based on instantiating function variables and discharging the associated assumptions. Pavlovic et al. [13,3] similarly refine protocols by transforming messages and propose specialized formalisms for establishing secrecy and authentication properties. Unfortunately, their approach lacks a formal semantics.

Guttman [9,8] studies the preservation of security properties by a rich class of protocol transformations in the strand space model. His approach to property preservation is based on the simulation of protocol analysis steps instead of execution steps. Each analysis step explains the origin of a received message. However, he does not provide syntactic conditions for the transformations to preserve security properties.

The objective of our work is to develop a comprehensive theory of protocol transformations covering a wide range of protocols and security properties. Our starting point is Hui and Lowe's work [10]. They work in a restricted protocol model with typed messages and atomic keys and show their results for ground messages. However, in order to transform protocol descriptions, we have to consider messages with variables and justify that a transformed attack is indeed an execution of the abstract protocol. They only discuss this important point briefly and informally. We plan to address these issues and obtain preservation results for relevant classes of security protocols (such as those based on convergent subterm theories [1]) and expressive property specification languages (such as PS-LTL [4] or the language proposed in [11]). We aim to cover a large class of protocol transformations including those described in the examples in [3,6,5].

We intend the following contributions. We want to significantly extend the scope of existing work in terms of expressiveness of the protocol specifications, the protocol transformations, and the preserved properties. Our work will provide a sound formal underpinning to protocol transformations, which can serve as a foundation for rigorous security protocol development, modifications, and comparisons of existing protocols.

2 A motivating example

We present the development of a key transport protocol as a motivating example. We state the protocols in standard Alice&Bob notation and describe each refinement step as a protocol transformation.

Consider a key transport protocol P_1, where a key server S generates and distributes a session key K_{AB} to an initiator A and a responder B.

M1.1. $A \rightarrow S : A, B$
M1.2. $S \rightarrow A : \{B, K_{AB}\}_{K_{AS}}$
M1.3. $S \rightarrow B : \{A, K_{AB}\}_{K_{BS}}$

In order to prevent replays and guarantee the recentness of K_{AB}, we refine this protocol into P_2 by adding a nonce and a timestamp to P_1.

M2.1. $A \to S : A, B, N_A$
M2.2. $S \to A : \{B, T_S, N_A, K_{AB}\}_{K_{AS}}$
M2.3. $S \to B : \{A, T_S, K_{AB}\}_{K_{BS}}$

Next, we obtain P_3 by refining the flow of protocol messages: the server now appends B's message in M2.3 to A's in M2.2, which A then forwards to B.

M3.1. $A \to S : A, B, N_A$
M3.2. $S \to A : \{B, T_S, N_A, K_{AB}\}_{K_{AS}}, \{A, T_S, K_{AB}\}_{K_{BS}}$
M3.3. $A \to B : \{A, T_S, K_{AB}\}_{K_{BS}}$

In P_3, B cannot be sure that A has received the key K_{AB}. We refine P_3 into P_4 by having the server encrypt B's message inside A's, which allows A to authenticate B on K_{AB}.

M4.1. $A \to S : A, B, N_A$
M4.2. $S \to A : \{B, T_S, N_A, K_{AB}, \{A, T_S, K_{AB}\}_{K_{BS}}\}_{K_{AS}}$
M4.3. $A \to B : \{A, T_S, K_{AB}\}_{K_{BS}}$

Protocol P_4 is a basic form of the Kerberos IV protocol (without authenticators). We have started from a simple initial protocol guaranteeing only the secrecy of the session key. We have then used refinement to add several features to this protocol in order to obtain key freshness, recentness, and authentication properties.

For additional examples of protocol developments, we refer the reader to [3,6,5].

3 Approach and current work

Security protocol model We briefly summarize our security protocol model, which is based on [11]. The model is parametrized by a message term algebra over a given signature Σ and a set of variables $\mathcal{V}$. Constants model nonces, keys, time stamps, and agents. Function symbols typically include hashes $h(t)$, pairs $\langle t, u\rangle$, and encryptions $\{t\}_u$. Let $\mathcal{T}$ be the set of all terms over Σ and $\mathcal{V}$. The terms may be quotiented by an equational theory, e.g., to model the commutativity of exponentiation for a Diffie-Hellman protocol. As is standard, we model a Dolev-Yao intruder [7] with full control over the network using a deduction system. Its judgements have the form $T \vdash u$, meaning that the intruder can derive the term u from a finite set of terms T. Encryption is perfect, i.e., the intruder can only decrypt with the intended key.

We specify protocols as finite sets of roles instead of the informal Alice&Bob notation from Section 2. Each role $R \in \mathit{Role}$ is a sequence of send and receive events of the form $\mathsf{snd}(t)$ or $\mathsf{rcv}(t)$ for a term t. The semantics of a protocol is a transition system with states of the form $s = (tr, th, \sigma)$, where tr is a trace consisting of a sequence of pairs of thread identifiers and events, $th : \mathit{TID} \rightharpoonup \mathit{Role}$ is a thread pool, and $\sigma = \{\sigma_i \mid i \in \mathit{dom}(th)\}$ is a family of ground substitutions σ_i, one for each thread i. The transitions are defined by an operational semantics with rules for sending and receiving messages. The receive rule includes a premise requiring that the received message is deducible by the intruder from his initial knowledge and the sent messages. We write $\mathcal{R}(P)$ for the set of reachable states of the protocol P.

Protocol transformations Our protocol transformations are functions $f\colon \mathcal{T} \to \mathcal{T}$ on terms, which we lift to events, roles, protocols, traces, and states. We consider a class of *nice* transformations, which includes the following operations on messages:

1. remove encryptions and hashes,
2. remove fields from an encrypted message,
3. pull fields outside of an encryption,
4. split encryption into several ones, and
5. project pairs (under certain conditions) and reorder pairs.

These protocol transformations simplify messages (and hence protocols) and can therefore be understood as abstractions. However, the same transformations can be used for protocol refinements, which proceed in the opposite direction, from abstract to concrete. For example, in Section 2, the refinement of P_1 into P_2 uses transformations of the second type, and the one from P_3 into P_4 uses a transformation of the third type.

So far we do not cover structural transformation of protocol like the message relaying transformation (cf. the refinement of P_2 to P_3), but we plan to do so in the future.

Property specification language We consider a property specification language with formulas of the following shape.

$$\phi = \forall i_1, \ldots, i_m. \bigwedge_{A \in \Gamma} A \;\Rightarrow\; \exists j_1, \ldots, j_n. \bigwedge_{B \in \Delta} B \tag{1}$$

The quantifiers range over thread identifiers and Δ, Γ are sets of atomic predicates. These predicates include $\mathit{learns}(m)$ for expressing intruder knowledge in secrecy properties, and event orderings $e \prec e'$ and equations $m = m'$ for authentication properties. To achieve attack preservation, the $\mathit{learns}(m)$ is only allowed to occur in Γ. A state $s = (tr, th, \sigma)$ that does not satisfy a property ϕ, written $s \not\models \phi$, is called an attack on ϕ.

Attack preservation Suppose we are given a class of security protocols, properties, and transformations such as those sketched above. The main result we want to achieve is the preservation of attacks on a property ϕ of protocol P to attacks on the transformed protocol $f(P)$ and property $f(\phi)$. We formalize this property as follows.

$$\begin{aligned} &\forall (tr, th, \sigma) \in \mathcal{R}(P).\ (tr, th, \sigma) \not\models \phi \\ &\quad \Rightarrow (f(tr), f(th), f(\sigma)) \in \mathcal{R}(f(P)) \wedge (f(tr), f(th), f(\sigma)) \not\models f(\phi) \end{aligned} \tag{2}$$

We decompose the proof of such results into two parts: the preservation of (i) executability (first conjunct) and (ii) attacks (second conjunct).

Executability The proof that for each reachable state (tr, th, σ) of P the transformed state $(f(tr), f(th), f(\sigma))$ is reachable in $f(P)$ is based on the following deducibility preservation result.

$$T\theta \vdash u\theta \Rightarrow f(T)f(\theta) \vdash f(u)f(\theta) \tag{3}$$

This follows from two simpler properties. The first one is a simpler version of (3).

$$T \vdash u \;\Rightarrow\; f(T) \vdash f(u) \tag{4}$$

The second one requires that f satisfies the following substitution property, that is, for all terms t and substitutions θ,

$$f(t\theta) = f(t)f(\theta) \tag{5}$$

In particular, since the operational semantics of receive events requires the deducibility of the received message from previously sent messages, we can use (3) to show that each receive event of P can be simulated by a corresponding receive event in $f(P)$.

Attack preservation Since secrecy is expressed in terms of deducibility of messages, we obtain the preservation of secrecy for free from the above. For other properties, like those expressible in the language sketched above, a separate proof is needed.

We have proved property (4) for all nice transformations. However, the substitution property (5) turns out to be quite restrictive. It rules out transformations that look more than one level into the term structure (such as, e.g., for splitting an encryption). Our initial solution restricts the set of substitutions to *simple* ones, whose range contains no composed terms. This set covers typed substitutions, which are (implicitly) used in [10]. We have proved (5) (hence *executability*) and *attack preservation* for this restricted setting and a subclass of nice transformations specified by pattern matching. Unfortunately, this solution rules out untyped variables such as those required for forwarding messages (cf. Section 2).

4 Planned work and conclusions

Generalizing the results An alternative solution is based on the observation that executability depends on constraints $T \vdash u$ where the terms in T stem from send events and u from a receive event. Therefore, a restricted form of (5) where t ranges over the set of terms in the protocol roles suffices for executability. Since this form of (5) is protocol-dependent, we cannot use induction to establish it. Instead, we need to formulate criteria to check that a protocol has this property. For attack preservation, the substitution property must also hold for the terms occurring the properties we are interested in.

A different approach could replace the substitution $f(\theta)$ in (3) by some θ'. The construction of such a θ' would require a stronger proof technique, possibly based on symbolic constraint reduction [12]. This approach produces non-ground substitutions as solutions of constraint systems. Therefore, we would have some freedom to derive different ground substitutions θ'.

Outlook on future work In a longer-term perspective, we plan to extend the scope of transformations in several directions. First, we want to cover structural transformations, which not only modify messages, but also events and roles (e.g., relaying messages; splitting, merging, and deleting events). Second, we would like to cover a larger class of protocols, in particular, by including equational theories (e.g., Diffie-Hellman exponentiation, convergent subterm theories [1]). Third, we intend to extend the property language to include additional properties such as forward secrecy and also consider stronger adversary models (e.g., compromising session keys and local states). Finally, we also plan to implement a tool that supports the definition and application of protocol transformations and the guarantee of their soundness.

Conclusions In this work, we study attack-preserving security protocol transformations. These can be used for the abstraction, the refinement, and the comparison of protocols. Therefore, we consider this technique as a useful complement to verification.

So far, we have defined a subclass of transformations and proved the preservation of attacks with respect to a particular security property language. We have discussed the problems that we have encountered and proposed possible solutions. We have also sketched our plans for future work.

Acknowledgements This work is partially supported by the EU FP7-ICT-2009.1.4 Project No. 256980, NESSoS: Network of Excellence on Engineering Secure Future Internet Software Services and Systems.

References

1. M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1-2):2–32, 2006.
2. A. Armando, R. Carbone, L. Compagna, J. Cuéllar, and M. L. Tobarra. Formal analysis of SAML 2.0 web browser single sign-on: breaking the SAML-based single sign-on for Google apps. In V. Shmatikov, editor, *FMSE*, pages 1–10. ACM, 2008.
3. I. Cervesato, C. Meadows, and D. Pavlovic. An encapsulated authentication logic for reasoning about key distribution protocols. In *CSFW '05: Proceedings of the 18th IEEE workshop on Computer Security Foundations*, pages 48–61, Washington, DC, USA, 2005.
4. R. Corin, S. Etalle, and A. Saptawijaya. A logic for constraint-based security protocol analysis. In *IEEE Symposium on Security and Privacy*, pages 155–168, 2006.
5. A. Datta, A. Derek, J. C. Mitchell, and D. Pavlovic. Abstraction and refinement in protocol derivation. In *Proc. 17th IEEE Computer Security Foundations Workshop (CSFW)*, pages 30–, 2004.
6. A. Datta, A. Derek, J. C. Mitchell, and D. Pavlovic. A derivation system and compositionl logic for security protocols. *Journal of Computer Security*, 13:423–482, 2005.
7. D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
8. J. D. Guttman. Transformations between cryptographic protocols. In P. Degano and L. Viganó, editors, *ARSPA-WITS*, volume 5511 of *LNCS*, pages 107–123. Springer, 2009.
9. J. D. Guttman. Security goals and protocol transformations. In *Theory of Security and Applications (TOSCA), an ETAPS associated event*, volume 6993 of *LNCS*. Springer, 2011.
10. M. L. Hui and G. Lowe. Fault-preserving simplifying transformations for security protocols. *Journal of Computer Security*, 9(1/2):3–46, 2001.
11. S. Meier, C. J. F. Cremers, and D. A. Basin. Strong invariants for the efficient construction of machine-checked protocol security proofs. In *Proc. 23th IEEE Computer Security Foundations Symposium (CSF)*, pages 231–245, 2010.
12. J. K. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *ACM Conference on Computer and Communications Security*, pages 166–175, 2001.
13. D. Pavlovic and C. Meadows. Deriving secrecy in key establishment protocols. In *Proc. 11th European Symposium on Research in Computer Security (ESORICS)*, pages 384–403, 2006.
14. C. Sprenger and D. Basin. Developing security protocols by refinement. In *Proc. 17th ACM Conference on Computer and Communications Security (CCS)*, pages 361–374, 2010.
15. C. Sprenger and D. Basin. Refining key establishment. Technical Report 736, Computer Science Department, ETH Zurich, Sept. 2011.

Parametric Attack Graph Construction and Analysis*

Leanid Krautsevich **

Department of Computer Science, University of Pisa
Largo Bruno Pontecorvo 3, Pisa 56127, Italy
Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche
Via G. Moruzzi 1, 56124 Pisa, Italy
krautsev@di.unipi.it

Abstract. We present the first steps towards an implementation of attack graph construction and analysis technique based on inference rules. In our model, XML credentials describe basic attacks to the system, then inference rules allow composition of new attacks. We aim at modifying previously developed algorithm for the analysis of transitive trust models to the analysis of attack graphs. Important peculiarity of our model is exploitation of c-semirings for evaluation of system security level. C-semirings allow an application of the same algorithms for an analysis of attack graphs regardless of what metric is selected for the evaluation.

1 Introduction

Analysis and improvement of security of modern computer systems is a challenging task because the systems are extremely complex and heterogeneous. Often the analysis of security is based on attack graphs. Frequently, methods of the analysis are system and context specific and require manual adjustments. Moreover, most of the methods provide their own basic metric as the result of the analysis. We aim at creating a method that allows automated analysis of system security and works with wide range of security metrics without changing the core algorithm. Using different metrics for the evaluation helps to provide different views on system security and allows a security administrator to judge better on improvements to security of a system.

The essential elements of our method are basic attacks described as XML credentials similar to RTML [10]. Basic attacks form an attack graph with the nodes representing sets of resources and the edges representing the attacks. All the edges are labelled with costs of attacks. We introduce three inference rules, that allows us to make conclusions on the system security. The rules are compliant with rules presented for reasoning on transitive trust models. Thus, we can adopt an earlier developed algorithm [11,5] for the analysis of the attack graphs.

* This paper was partly supported by EU-FP7-ICT NESSoS project.
** Supervisor: Fabio Martinelli, Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Pisa, Italy, e-mail: fabio.martinelli@iit.cnr.it.

We assume, that costs of attacks stand for security metrics, used for the evaluation of the system security. We associate each security metric with c-semiring which is algebraic structure used for the analysis of weighted graphs, e.g., for searching a shortest path in a graph. C-semirings allow to create an algorithm for the analysis of attack graphs that does not depend on the security metric selected for the analysis of the system.

1.1 Contributions

Main contributions of the paper are the following:

- the method for the analysis of attack graphs is based on inference rules similar to ones used for the analysis of transitive trust models, thus, the method may reuse the slightly changed algorithm developed for the analysis of transitive trust models;
- the method works regardless of the security metric selected for the evaluation due to the use of c-semiring algebraic structure.

The rest of the paper is structured as follows. Section 2 describes an application scenario and introduces inference rules. Section 3 discusses the exploitation of XML for representing basic attacks and introduces XML based rules for the processing of attacks. Section 4 observes the related work and Sect. 5 provides the conclusion and the future work.

2 Application Scenario

We consider a scenario where a security administrator performs the evaluation of security on the basis of resources available to an attacker. The features and the notation of the model: $ATT = \{a_1, \ldots, a_m\}$ is a set of attacks to a system, $RES = \{r_1, \ldots, r_n\}$ is a set of resources in the system, $S = \{a_1 \ldots a_k \mid a \in ATT\}$ is an attack sequence, R is a set of resources available to the attacker, G is a set of resources gained as the result of an attack, w is a cost of the attack, W is a cost of the attack sequence.

There is a set of basic attacks that can be applied when the attacker has an initial set of resources. The attacker obtains new resources by applying an attack. In our model, the resources are not consumed and the resources that can not be reached are not taken into account. We also consider the sequential composition of attacks, i.e., the attacker can perform attacks one by one. Moreover, all the attacks have costs, thus, all the potential resources are reachable with the corresponding costs. The attacker selects the attack with the best cost, e.g., the highest probability of success.

We introduce two operators $\otimes, \oplus$ over some domain D of values of costs, where the former operator serves for aggregation of costs of attacks in a sequence and the latter operator for the selection of the attack with the better cost. For example, the operator $\otimes$ equals $\times$ (multiplication), $\oplus$ equals max that stands for

the selection of the attack sequence with the maximal probability of success, and the domain is $D = [0, 1]$. We can extend this basic set of operators to couples $(sequence, cost)$. Suppose, there are sequences of attacks a_1, a_2 with costs w_1, w_2:

$$(a_1, w_1) \otimes' (a_2, w_2) = (a_1 a_2, w_1 \otimes w_2)$$

$$(a_1, w_1) \oplus' (a_2, w_2) = \begin{cases} (a_1, w_1) & \text{if } (w_1 \oplus w_2) = w_1 \\ (a_2, w_2) & \text{if } (w_1 \oplus w_2) = w_2 \end{cases}$$

where $a_1 a_2$ is an order preserving concatenation of attacks.

Now we are ready to present three inference rules that allow us to analyse the above model.

First, we consider a set of resources available, say R_X. By starting from this set of resources, an intruder can perform a basic attack that simply needs a subset R_i of these resources and then acquires new resources G_j. This is modelled by the ***basic attack*** rule

$$\frac{R_i \xrightarrow{(a_q, w_q)} R_j \quad R_i \subseteq R_X}{R_X \xrightarrow{(a_q, w_q)} R_t} \tag{1}$$

where $R_j = R_i \cup G_j$ and $R_t = R_X \cup G_j$.

Then it is possible to compose several different basic attacks in a sequence and this is done by the ***composite attack*** rule. It states that starting from a set of resources by applying an attack the intruder gets new resources that serve as a basic set for another attack. Thus, a sequence of attacks is built.

$$\frac{R_i \xrightarrow{(a_q, w_q)} R_j \quad R_j \xrightarrow{(a_p, w_p)} R_k}{R_i \xrightarrow{(a_q, w_q) \otimes' (a_p, w_p)} R_k} \tag{2}$$

Finally, the ***attack selection*** rule selects the attack with the better cost.

$$\frac{R_i \xrightarrow{(a_q, w_q)} R_j \quad R_i \xrightarrow{(a_p, w_p)} R_j}{R_i \xrightarrow{(a_q, w_q) \oplus' (a_p, w_p)} R_j} \tag{3}$$

Rules 2 and 3 may be generalized for an application to attack sequences by using S and W instead of a and w.

The analysis of a system works as follows. Starting from the initial set of basic attacks, we build a graph whose nodes are sets of resources R and which arcs are labelled with attack costs. We need to apply the rules and to consider all the sequences exiting from the initial set R_i to the state $R_i \cup G_k$ and which cost is better than a total cost W. The overall protection goal can be to avoid the attacker to control the set of resources $R_i \cup G_k$ with the total cost better than the total cost W.

We propose to present costs as a special mathematical structure *c-semiring* (constraint semiring) [4]:

Definition 1. C-semiring T *is a tuple* $\langle D, \oplus, \otimes, \mathbf{0}, \mathbf{1} \rangle$*:*

- *D is a set of elements and $\mathbf{0}, \mathbf{1} \in D$;*
- *$\oplus$, is an additive operator defined over (possibly infinite) set of elements D, for $d_1, d_2, d_3 \in T$, it is commutative ($d_1 \oplus d_2 = d_2 \oplus d_1$) and associative ($d_1 \oplus (d_2 \oplus d_3) = (d_1 \oplus d_2) \oplus d_3$), and $\mathbf{0}$ is a* unit *element of the additive operator ($d_1 \oplus \mathbf{0} = d_1 = \mathbf{0} \oplus d_1$).*
- *$\otimes$ is a binary multiplicative operator, it is associative and commutative, $\mathbf{1}$ is its* unit *element ($d_1 \otimes \mathbf{1} = d_1 = \mathbf{1} \otimes d_1$), and $\mathbf{0}$ is its* absorbing *element ($d_1 \otimes \mathbf{0} = \mathbf{0} = \mathbf{0} \otimes d_1$);*
- *$\otimes$ is distributive over additive operator ($d_1 \otimes (d_2 \oplus d_3) = (d_1 \otimes d_2) \oplus (d_1 \otimes d_3)$);*
- *$\leq_T$ is a partial order over the set D, which enables comparing different elements of the semiring, the partial order is defined using the additive operator $d_1 \leq_T d_2$ (d_2 is better than d_1) iff $d_1 \oplus d_2 = d_2$ (idempotence).*

For a security metric, we need to determine the domain of values D and two operators $\oplus$ and $\otimes$ that are further used for the analysis of an attack graph. An example may be *shortest attacks path* metric and c-semiring with $\oplus$ equals min, $\otimes$ equals summation, and the domain D is the set of natural numbers $\mathbb{N}$. Other c-semirings may be defined for other metrics.

3 Using XML Credential to Represent and Reason on Attacks

We use XML credentials to store the information about basic attacks. Basic attacks are used to compute composite attacks sequences. Composite attacks are also represented by XML credentials and are used when necessary. XML credentials allow us to use slightly modified algorithm for dealing with trust relationships for access control systems [11,5] to deal with attack graph. Thus, we use two kinds of credential: one for modelling a basic attack b, and another one for modelling a composed attack c, where an attacker is A.

In case of a ***basic attack***, a is a sequence which contains only a single attack, R is the minimal resources necessary to perform the attack, G is the set of gained resources and w is the cost of the attack:

$$A.b(a, R, G, w) \tag{4}$$

In case of a ***composite attack***, S is a sequence of attacks, R represents the initial set of resources, F is the final set of resources and W the cost of the attack sequence S.

$$A.c(S, R, F, W) \tag{5}$$

Instantiations of Equations 1, 2, 3 for XML credentials are the following.

$$\frac{A.b(a, R, G, w) \quad R \subseteq X}{A.c(a, X, X \cup G, w)} \tag{6}$$

$$\frac{A.c(S_1, R_1, F_1, W_1) \quad A.c(S_2, R_2, F_2, W_2) \quad R_2 \subseteq F_1}{A.c(S_1 S_2, R_1, F_2, W_1 \otimes W_2)} \tag{7}$$

$$\frac{A.c(S_1, R, F, W_1) \quad A.c(S_2, R, F, W_2)}{A.c(S_1 \odot S_2, R, F, W_1 \oplus W_2)} \tag{8}$$

where S_1S_2 is a concatenation of attack sequences, $\odot$ corresponds to the selection of sequence with the better cost. Now we can adopt algorithm [5] to the analysis of attack graphs since the rules are similar to rules [5,11] for reasoning on trust.

4 Related Work

The attacker model we use in the paper could be seen as an attack graph [1,13,12,6]. E.g., in [1] a (constrained) graph model based on resource acquisition by the attacker has been developed, the model considers the local knowledge of the attacker stored in nodes during the attack-path analysis (also for the selection of countermeasures).

Different security metrics are used for analysis of attack graphs: probability of successful attack [15], minimal cost of attack [14], minimal cost of reduction [16], shortest path [13]. Some of these metrics could be seen as specific instance of semirings, thus also suitable for the analysis with out approach. On the other hand, our approach is parametric and can also use other metrics for the analysis.

Krautsevich et al., [7] formally modelled and defined several security metrics which measure security system out of the context. The metrics were analysed in order to check if some of them provide the same evaluation. The next step in this study was establishing relations between these metrics and risk [8]. Every metric study was considered separately, when our current work is more generic.

To our knowledge, there are several attempts of applying semirings in security area [2,3]. The authors used semirings for the analysis of integrity policies, cryptographic protocols, and computation of trust levels through trust chains. Krautsevich et al., [9] applied semirings to analysis of security of process-like structures for describing web services. In this work, we provide a wider range of application of semirings for security analysis.

5 Conclusion

We used XML credentials to describe basic attacks and proposed inference rules for composition and selection of the attacks. C-semiring allows us to make the method independent of what security metric is selected for the evaluation. Finally, we worked towards an adoption of existed algorithm for reasoning on transitive trust to the analysis of parametric attacks graphs.

As a future work, we would like, first, to introduce modified algorithm for the analysis of attack graphs. Second, we would like to extend our approach for other models of attack graphs, e.g., privileges graph. Moreover, we would like to implement our method as a software prototype and perform an analysis the properties of the method, e.g, performance. For the implementation, we plan to minimally modify the code of algorithm for evaluation of RTML credentials with semirings developed in [5].

References

1. F. Baiardi, F. Martinelli, L. Ricci, and C. Telmon. Constrained automata: a formal tool for risk assessment and mitigation. *Journal of Information Assurance and Security*, 3:304–312, 2008.
2. G. Bella, S. Bistarelli, and S. N. Foley. Soft constraints for security. In *Proceedings of the First International Workshop on Views on Designing Complex Architectures (VODCA '04)*, volume 142 of *Electronic Notes in Theoretical Computer Science*, pages 11–29. Elsevier, 2006.
3. S. Bistarelli, F. Martinelli, and F. Santini. A semantic foundation for trust management languages with weights: An application to the rt family. In *Proceedings of the 5th international conference on Autonomic and Trusted Computing*, ATC '08, pages 481–495, Berlin, Heidelberg, 2008. Springer-Verlag.
4. S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint satisfaction and optimization. *J. ACM*, 44(2):201–236, March 1997.
5. D. Fais, M. Colombo, and A. Lazouski. An implementation of role-base trust management extended with weights on mobile devices. In *Proceedings of the 4th International Workshop on Security and Trust Management*, volume 244 of *Electronic Notes in Theoretical Computer Science*, pages 53–65. Elsevier, 2009.
6. S. Jha, O. Sheyner, and J. M. Wing. Minimization and reliability analyses of attack graphs. Technical Report CMU-CS-02-109, Carnegie Mellon University, 2002.
7. L. Krautsevich, F. Martinelli, and A. Yautsiukhin. Formal approach to security metrics. what does "more secure" mean for you? In *Proceedings of the 1st International Workshop on Measurability of Security in Software Architectures*, 2010.
8. L. Krautsevich, F. Martinelli, and A. Yautsiukhin. Formal analysis of security metrics and risk. In *Proceedings of the IFIP Workshop on Information Security Theory and Practice*, volume 6633, pages 304–319. 2011.
9. L. Krautsevich, F. Martinelli, and A. Yautsiukhin. A general method for assessment of security in complex services. In *Proceedings of 4th European Conference ServiceWave*. Springer, 2011.
10. N. Li, J. C. Mitchell, Y. Qiu, W. H. Winsborough, K. E. Seamons, M. Halcrow, and J. Jacobson. Rtml: A role-based trust-management markup language. Technical report, Purdue University, 2004.
11. F. Martinelli and M. Petrocchi. On relating and integrating two trust management frameworks. In *Proceedings of the Second International Workshop on Views on Designing Complex Architectures (VODCA '06)*, volume 168 of *Electronic Notes in Theoretical Computer Science*, pages 191–205. Elsevier, 2007.
12. S. Noel and S. Jajodia. Managing attack graph complexity through visual hierarchical aggregation. pages 109–118, New York, NY, USA, 2004. ACM Press.
13. R. Ortalo, Y. Deswarte, and M. Kaaniche. Experimenting with quantitative evaluation tools for monitoring operational security. 25(5):633–650, 1999.
14. J. Pamula, S. Jajodia, P. Ammann, and V. Swarup. A weakest-adversary security metric for network configuration security analysis. In *QoP '06: Proceedings of the 2nd ACM workshop on Quality of protection*, pages 31–38. ACM Press, 2006.
15. L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia. An attack graph-based probabilistic security metric. In *Proceeedings of the 22nd annual IFIP WG 11.3 working conference on Data and Applications Security*, pages 283–296, Berlin, Heidelberg, 2008. Springer-Verlag.
16. L. Wang, S. Noel, and S. Jajodia. Minimum-cost network hardening using attack graphs. *Journal Computer Communications*, 29(18):3812–3824, 2006.

Enhancing Safety and Security of Distributed Systems through Formal Patterns*

Jonas Eckhardt[1,2,3], Tobias Mühlbauer[1,2,3]
Supervisors: José Meseguer[4], Martin Wirsing[1]

[1] Ludwig Maximilian University of Munich, [2] Technical University of Munich,
[3] University of Augsburg, [4] University of Illinois at Urbana-Champaign

Abstract. Distributed systems are often safety- and security-critical systems and have strong qualitative and quantitative formal requirements, equally important time-critical performance-based quality of service properties, and need to dynamically adapt to changes in a potentially hostile and often probabilistic environment. These aspects make distributed systems complex and hard to design, build, test, and verify. To tackle this challenge, we propose a formal pattern-based approach and framework for the design of correct-, secure-, and safe-by-construction distributed systems.

Key words: formal patterns, meta-object pattern, statistical model checking, rewriting logic, distributed systems, cloud computing

1 Introduction

On June 20, 2011, the Cloud-based file storage service Dropbox reported that "Yesterday we made a code update at 1:54pm Pacific time that introduced a bug affecting our authentication mechanism. We discovered this at 5:41pm and a fix was live at 5:46pm." [4]. During these nearly four hours, the broken authentication mechanism granted access to possibly private data stored on some accounts using any chosen password. Issues like this are not the exception which is also reflected by the list of the top 10 obstacles for the adoption and growth of Cloud Computing [5]; with data confidentiality and auditability, availability of service, and bugs in large distributed systems being obstacles on this list. In fact, distributed systems (i) are safety- and security-critical systems which have strong qualitative and quantitative formal requirements, (ii) have equally important time-critical performance-based quality of service properties (e.g., availability), and (iii) need to dynamically adapt to changes in a potentially hostile (e.g., distributed denial of service attacks) and often probabilistic environment they operate in. These aspects make distributed systems complex and hard to design, build, test, and verify.

Modular approaches tackle the aforementioned complexity in the early stages of system design and analysis. These approaches include design patterns, which are general, reusable solutions to commonly occurring software problems and

* This work has been partially sponsored by the Software Engineering Elite Graduate Program and the EU-funded project FP7-256980 NESSoS.

have been successfully used in different domains including object-oriented software design [10], service-oriented computing [12,9] and security [16]; they clearly define the programming context, the problem, and the advantages and disadvantages of the design solution (see e.g., [10,16]).

In addition to "normal" design patterns, formal patterns are reusable solutions that are formally specified with precise semantic requirements and come with strong formal guarantees. Distributed systems can be specified as compositions of instances of such formal patterns.

Research Goals and Contributions. The main goal of the proposed research is to contribute a formal pattern-based approach and framework for the design of correct-, secure-, and safe-by-construction distributed systems, aided by a rich tool environment. The approach is based on the ideas of (i) developing executable formal models of distributed system designs, (ii) making these designs modular based on highly reusable formal patterns, and (iii) formally analyzing such models to verify qualitative (e.g., invariants) and quantitative (e.g., expected throughput) properties. This approach distinguishes itself by using executable formal pattern-based system specifications and statistical model checking, which allows the verification of larger system instances than with conventional model checking techniques (state explosion).

Rewriting logic and Maude. In order to specify executable formal patterns, an appropriate semantic framework is needed. We chose rewriting logic [13], a simple, yet powerful, computational logic and a general formalism that is a natural model of computation and an expressive semantic framework for concurrency, parallelism, communication, interaction, and object-orientation. It is capable of logical and distributed object reflection and, through its probabilistic [2] and real-time extensions [15], of modeling real-time, stochastic, and hybrid systems.

Maude [7] is a high-performance implementation of rewriting logic capable of executing rewriting logic-based specifications. The key concept of Maude specifications is that of a module. *Object-oriented modules* define objects, their state, and messages; where objects communicate via asynchronous or synchronous message passing. Distributed systems are modelled by object-oriented modules, where the state of such a system is a multiset or "soup" of objects and messages, called a *configuration*. A *parameterized module* $M[X :: P]$ has a formal parameter X satisfying a parameter theory P; M can be instantiated by another module Q via a theory interpretation $V : P \longrightarrow Q$, called a *view*, with the usual pushout semantics (see [7]). We denote the resulting module by $M[V]$ or shorter by $M[Q]$ if V is clear from the context.

The Maude system has an extensive tool environment which, e.g., includes a LTL model checker (see [7]) and the statistical model checker PVeStA [3]. The Maude system and its tool environment are the foundation of our work.

Outline. This paper proceeds as follows: Sect. 2 introduces the concept of formal patterns and gives the example of the general meta-object pattern. In Sect. 3 we discuss our proposed approach for the design of correct-, secure-, and safe-by-construction distributed systems in more detail. In Sects. 4 and 5, we respectively discuss a research plan for future work and summarize our results.

2 Formal Patterns

Formal patterns [8] enhance pattern descriptions with formal executable specifications that can support the mathematical analysis of qualitative and quantitative properties. Just as "normal patterns", a formal pattern Pat is structured in context, problem, solution, advantages and shortcomings (cf. e.g. [16,9]). Instead of using UML or Java we describe these patterns formally as a paramterized module $M[S]$ with a parameter theory S in Maude. The context of the pattern typically includes a description of the assumptions of the parameter theory S. Many of the advantages and shortcomings of the formal pattern can be gained from formal analyses.

Two formal patterns Pat and Pat' can be composed by the pattern composition $Pat + Pat'$. The problem statement and context of $Pat + Pat'$ can be systematically derived from those of Pat and Pat'. As we will see, such a composition of patterns might combine advantages while cancelling out disadvantages.

Example: The Meta-Object Pattern. Many distributed systems need to function in potentially hostile environments such as the Internet. Additionally, safety, real-time and quality of service requirements need to be satisfied. Modularization is an instrument that helps the designer or architect to cope with the high complexity of such a system. The Meta-Object (MO) pattern provides an approach based on modularization. It is defined as follows:

Context. A concurrent and distributed object-based system.

Problem. How can the communication behavior of one or several objects be dynamically *mediated/adapted/controlled* for some specific purposes?

Solution. A meta-object is an object which dynamically *mediates/adapts/controls* the communication behavior of one or several objects under it. In rewriting logic, a meta-object can be specified as an object with an inner configuration that contains the object or objects that are controlled by the meta-object. Thus, the parameterized module $MO[X]$ introduces the meta-object constructor; the parameter X specifies the controlled system.

Advantages and Shortcomings. MO defines a general control and wrapper architecture; but may add communication indirection and the requirement for language specific object visibility.

MO is a widely used pattern: The meta-object is sometimes called an *onion-skin* meta-object [1] if the inner configuration contains a single object, which itself could be wrapped inside another meta-object, and so on, like the skin layers in an onion. More generally, the inner configuration may not only contain several objects $o_1 \ldots, o_m$ inside: it may also be the case that some of these o_i are themselves meta-objects that contain other objects, which may again be meta-objects, and so on. That is, the more general reflective meta-object architectures are so-called "Russian Dolls" architectures [14].

Figure 1 illustrates the idea of a hierarchical composition of meta-objects and components according to the Russian Dolls model with boundary-crossing messages M, M', and M''. Messages addressed to the internal components $C_1 \ldots C_N$ first need to cross the boundaries of the two outer meta-objects MO_1 and MO'_2.

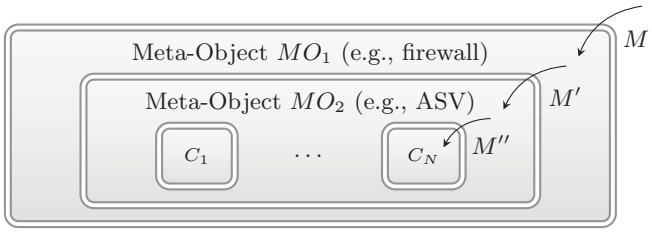

Fig. 1. Example of a hierarchical composition of meta-objects and components according to the Russian Dolls model with boundary-crossing messages.

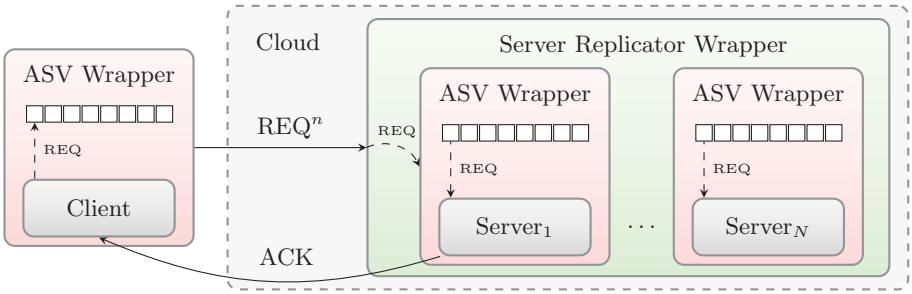

Fig. 2. Application of the ASV$^+$SR meta-object composition on a Cloud-based client-server request-response service.

The outermost meta-object MO_1 may thereby be a firewall that forwards selected messages to its inner configuration according to specific filter rules, and the inner meta-object MO_2 may be a **D**istributed **D**enial **o**f **S**ervice (DDoS) defense mechanism like the ASV protocol [11].

3 Approach: Enhancing Safety and Security through Formal Patterns

In [8], two additional formal patterns, the **S**erver **R**eplicator (SR) and the ASV pattern, are introduced. In cases of high demand (e.g., a raising number of requests or a DDoS attack), the SR pattern replicates instances of a parametric server on demand while the ASV pattern represents a modularized specification of the ASV protocol, which provides a defense mechanism against DDoS attacks for a parametric client-server request-response system. Under DDoS attacks, the goal is to provide stable availability, i.e., that with very high probability service quality remains very close to a threshold, regardless of how bad the DDoS attack can get. Quantitative analysis of the two patterns has shown that the ASV pattern does not provide stable availability and that the SR pattern cannot provide stable availability at a reasonable cost. However, for the composition of ASV and SR, ASV^+SR (see Fig. 2), it has been shown that stable availability at a reasonable cost can be achieved.

Based on this example, we propose a general approach for enhancing safety and security of distributed systems through formal patterns:

1. Develop executable formal models of distributed systems in rewriting logic, supported by the Maude system.
2. Make these specifications modular and adaptive using instances of formal patterns. A catalog thereby provides highly reusable formal patterns such as the Meta-Object, ASV, and SR patterns.
3. Formally analyze these models to verify qualitative and quantitative properties using the Maude tool environment (e.g., parallelized statistical model checking supported by PVeStA is able to analyze large system models).
4. Identify reusable formal patterns in the model and add their formal specifications to the pattern catalog.

4 Research Plan for Future Work

The main goal of the proposed research is to contribute a formal pattern-based approach and framework for the design of correct-, secure-, and safe-by-construction distributed systems, aided by a rich tool environment. We propose three main areas of future research: (i) build a rich and comprehensive catalog of formal patterns, (ii) identify security, safety, and other properties that are preserved by pattern composition and proof their preservation, and (iii) improve the existing tool support.

To build a rich and comprehensive catalog of formal patterns, existing patterns that are not yet explicitly modelled as a formal pattern need to be identified and formally specified.

In [6], it has been shown that the cookies protocol (a DDoS defense protocol), if wrapped around a system, preserves the safety properties of the wrapped system. We conjecture that the same is true for the ASV and $ASV^{+}SR$ protocols. In a first step, we want to prove that the ASV protocol also retains safety properties of the wrapped client-server request-response system. In the future, we want to identify such properties of other patterns and prove that they are preserved when the pattern is applied to a system. Having property preserving formal patterns improves their composability and reduces the formal verification effort as specific properties are, by construction, preserved in the composed model.

Furthermore, we propose to improve existing tool support in two main areas: (a) the robustness of existing tools and (b) code generation from executable formal models. Since we want to build systems in which many participants are communicating with each other and perform quantitative analyses on such systems, we need analysis and verification tools that scale with the size of these systems. In particular, analysis tools such as the PVeStA [3] statistical model checker, which drastically increases the scalability of statistical model checking through parallelization, need to be improved in terms of fault tolerance. Finally, to incorporate the proposed approach in an software engineering process, code generation techniques are needed. Thereby, based on correct-, secure-, and safe-by-construction specifications, correct-, secure-, and safe-by-construction implementations are generated.

5 Conclusion

In this paper we have presented the research goal of a formal pattern-based approach and framework for the design of correct-, secure-, and safe-by-construction distributed systems, aided by a rich tool environment. We gave a description of formal patterns, including the example of the Meta-Object pattern, and gave references to existing work that shows that formal patterns can help deal with security and safety issues and that formal analysis can help evaluate patterns in various contexts. In particular, we gave a description of the general formal pattern-based approach and concluded this paper with a summary of a research plan for future work.

References

1. G. Agha, S. Frolund, R. Panwar, and D. Sturman. A Linguistic Framework for Dynamic Composition of Dependability Protocols. *IEEE ICPADS*, 1:3–14, 1993.
2. G. Agha, J. Meseguer, and K. Sen. PMaude: Rewrite-based specification language for probabilistic object systems. *ENTCS*, 153(2):213–239, 2006.
3. M. AlTurki and J. Meseguer. PVeStA: A parallel statistical model checking and quantitative analysis tool. In *CALCO*, volume 6859 of *LNCS*, pages 386–392, 2011.
4. Arash Ferdowsi. Yesterday's Authentication Bug. `http://blog.dropbox.com/?p=821` (01/2012).
5. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the Clouds: A Berkeley View of Cloud Computing. Technical report, University of California at Berkeley, 2009.
6. R. Chadha, C. A. Gunter, J. Meseguer, R. Shankesi, and M. Viswanathan. Modular Preservation of Safety Properties by Cookie-Based DoS-Protection Wrappers. In *FMOODS*, volume 5051 of *LNCS*, pages 39–58, 2008.
7. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. *All About Maude - A High-Performance Logical Framework: How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350 of *LNCS*. Springer, 2007.
8. J. Eckhardt, T. Mühlbauer, M. AlTurki, J. Meseguer, and M. Wirsing. Stable Availability under Denial of Service Attacks through Formal Patterns. In *FASE*, volume 7212 of *LNCS*, 2012.
9. T. Erl. *SOA Design Patterns*. Prentice Hall, 2008.
10. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.
11. S. Khanna, S. Venkatesh, O. Fatemieh, F. Khan, and C. Gunter. Adaptive Selective Verification. In *IEEE INFOCOM*, pages 529–537, 2008.
12. M. Wirsing et al. Sensoria Patterns: Augmenting Service Engineering. In *ISoLA*, volume 17 of *CCIS*, pages 170–190, 2008.
13. J. Meseguer. Conditional Rewriting Logic as a Unified Model of Concurrency. *TCS*, 96(1):73–155, 1992.
14. J. Meseguer and C. Talcott. Semantic models for distributed object reflection. In *ECOOP*, volume 2374, pages 1–36. LNCS, 2002.
15. P. C. Ölveczky and J. Meseguer. Semantics and pragmatics of Real-Time Maude. *HOSC*, 20(1–2):161–196, 2007.
16. M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad. *Security Patterns*. Wiley, 2005.

Service-Oriented Trust and Reputation Architecture *

Francisco Moyano

Department of Computer Science, University of Malaga, 29071, Málaga, Spain
{moyano,mcgago,jlm}@lcc.uma.es

Abstract. As the Future Internet arrives, more complex, service-based applications are spreading. These applications pose several challenges, including the huge amount of entities that must interact and their heterogeneity. The success of these applications depends on the collaboration and communication of these entities, that might belong to different organizations and administrative domains. Therefore, trust and reputation become two crucial issues. We propose the specification and design of a service-based security architecture that stresses the delivery of trust and reputation services to any application that might require them.

Supervisors: Carmen Fernandez-Gago and Javier Lopez

1 Introduction: Problem and Motivation

The context that frames this work is that of Service-Oriented Architectures, which is described in Section 1.1. The problem that we mean to address and the motivation are presented in Section 1.2.

1.1 A Brief Introduction to Service-Oriented Architectures

A rather complete definition of Service-Oriented Architecture (SOA) is "the policies, practices, frameworks that enable application functionality to be provided and consumed as sets of services published at a granularity relevant to the service consumer. Services can be invoked, published and discovered, and are abstracted away from the implementation using a single, standards-based form of interface" [17]. In general, a service implements a limited and specific functionality, and can be discovered and called by means of standard technologies.

SOA is also often understood as being an architectural style that defines several components with their relations and constraints from which many different

* The research leading to these results have received funding from the European Community's Seventh Framework Programme FP7/2007-2013 as part of the Network of Excellence NESSoS (www.nessos-project.eu) under grant agreement number 256980. The first author is funded by the Spanish Ministry of Education through the National F.P.U. Program.

architectures can be derived. However, architects of service-oriented systems often find conflicts when trying to reconcile the business goals, the non-functional requirements, and some principles that are a consequence of the SOA approach, such as standardization at multiple levels, loose coupling, reusability, composability, and discoverability [6].

A crucial component for many SOAs is the Enterprise Service Bus (ESB). An ESB is a complex piece of software that mediates between clients and services, and among services. It is an information bus that connects and allows the communication of heterogenous applications (e.g. Java Message Service (JMS) applications and Simple Object Access Protocol (SOAP) applications). It also caters for other commonly needed services, such as security, protocol conversion or exception handling.

1.2 Trust and Reputation in Service-Oriented Architectures for Future Internet Applications

SOA systems have been traditionally secured at the endpoints. This means that a service is hard-coded with security functions that realize security services. This approach has several shortcomings. One of them is the business and security coupling, which in turn makes manageability and interoperability more difficult. Reusability is also undermined, since although the service might fulfil a business goal interesting for the organization, the coupling with security would make it useless for the organization if the security policy of the service does not match that of the organization.

The Future Internet entails the arrival of new, complex service-based applications that span across multiple boundaries and administrative domains. This calls for trust and reputation services that assure the trustworthiness of the entities that take part of such heterogenous communications and collaborations, and which aid the decision-making processes. Examples of applications are those described in NESSoS [2]. These applications require the interaction of multiple entities and the management of personal and private information, therefore they are security-critical.

Trust and reputation services might assist traditional security services, such as encryption and authorization. The latter mechanisms can provide a trusted environment, where communication between service providers and clients is protected, and where access to business assets is limited. These mechanisms, if well leveraged, might provide certain degree of trust between clients and customers in general. However, the notions of trust and reputation focus on specific relationships between a given provider and a client, taking into account both local information (e.g. interaction records), and external information (e.g. recommendations from other clients or providers). Thus, traditional security mechanisms should provide the basis, the trusted medium onto which to build trust and reputation solutions.

Although there are some proposals towards the delivery of security as services, to the best of our knowledge none of them cope with trust and reputation services. Hafner et al. [10] propose an Enterprise Service Bus (ESB)-

based message-oriented Security as a Service architecture. Aurel [4] proposes SOSA (Service-Oriented Security Architecture), a security architecture for distributed web applications that mediates between a service requester and a service provider. WS-* security standards, such as WS-Security, WS-Trust, WS-Policy or WS-SecureConversation deal with confidentiality, integrity, non-repudation, and policy specification.

2 Aim: Trust and Reputation as a Service

As we mentioned in Section 1.2, the arrival of Future Internet demands trust and reputation solutions. In the complex, heterogeneous scenarios that arise in this context, trust mechanims must be leveraged since they play a crucial role in decision-making processes.

Trust is a complex concept for which a clear, standard definition has not been provided yet. A possible definition is the level of confidence that an entity of a system places on another entity of the same system for performing a given task. Thus, two features of trust are uncertainty and subjectivity. Reputation, on the other hand, is a more objective concept, and is based on information about or observations of the past behaviour of an entity. Both concepts are very related, and in fact, reputation can be used as a means to determine whether an entity can trust another entity [12].

Our aim is the specification and design of a functional, reusable service-based security architecture that emphasizes the delivery of trust and reputation services.

A key feature of this architecture must be its reusability, since it should serve as a framework for building multiple applications that require trust and reputation services, as it is the case of Future Internet applications. It is also important to note that we intend to analyze the relationship between trust, reputation, and other security services. The architecture might therefore deliver those security services that have an influence on the overall trust of any application.

In the literature there are important contributions in the field of trust and reputation that might assist our work.

Kiefhaber et al. [16] propose the Trust-Enabling Middleware, which provides generic funtionality for applications running on top of it that need to save, interpret, and query trust related information. However, the authors are oblivious of other security mechanisms that might have an important impact on the overall trust of the system.

It is also important for a reusable architecture to offer flexible mechanisms to accommodate or compose different trust models according to the needs of the application. In this direction, Huynh [7] proposed his Personalized Trust Framework (PTF), a rule-based system that makes use of semantic technologies for, given a domain, to apply the most suitable trust model. In a similar direction, Suryanarayana et al. [18] present PACE (Practical Architectural approach for Composing Egocentric trust), an architectural style for composing different trust models into the architecture of a decentralized peer in a P2P architecture.

3 Research Methodology

The methodology that we intend to follow to conduct the research consists of the following phases:

1. Exploratory phase: during this phase, a wide research will be done in order to gain a solid knowledge about the process and tools for architecting a service-oriented architecture.
2. Construction phase: the specification and design of the architecture will be done during this phase. In order to accomplish these tasks, we will accommodate the output of the previous phase, which provides us with a set of tools, methods and processes that currently exist to architect service-oriented solutions. This output could contain as well some found gaps that we will need to bridge during this phase.
3. Validation phase: NESSoS scenarios represent some of the most important Future Internet applications. Thus, the architecture will be validated for, at least, one of these scenarios, which are within the scope of e-Health or Smartgrids.

At the moment, we are at the exploratory phase, where we are surveying existing approaches towards the building of a service-based security architecture. Some of our findings are briefly described in the next section.

3.1 Processes and Tools for Architecting SOA Solutions

In general, the process for building any architecture is iterative and consists of several steps [9]. The first step is requirements elicitation, where the functional and non-functional requirements are identified and documented from the scenarios and the stakeholders. Next it is important to identify those requirements that drive the architecture building, namely the quality attributes (e.g. security, modifiability or performance) and constraints (e.g. reusability of legacy systems). These architectural requirements are prioritized. Trade-offs analysis might be required since requirements are often in conflict between each other. After choosing the set of architectural requirements, the architecture design takes place, when choosing a set of interrelated design patterns can help the architect. Finally, the architecture is validated. This validation consists of using scenarios to check by hand how the architecture responds to different stimulus. Another validation technique is early prototyping. In either ways, if a flaw is discovered, all the previous phases should be revised, leading to another iteration.

This generic process is also applicable to the case of SOA. Yet there are some more specific SOA-oriented processes, as the Rational Unified Process for Service-Oriented Modeling and Architecture (RUP SOMA) proposed by IBM [20], the Service-Oriented Modeling Framework, proposed by Michael Bell [5], and the SOA/TOGAF standard by The Open Group [3]. We will survey these processes to find one that better adapts to our needs.

The modeling of a service-oriented architecture requires designing many aspects, ranging from the business activities to the non service-oriented, existing

assets. However, the most interesting for us is the service model, where both the atomic and composite services are defined in terms of their specifications, interfaces, inputs, outputs, and collaborations. As with traditional object or component-based applications, services can be graphically modeled with the Unified Modelling Language (UML). Several UML profiles exist in the literature that allow modeling service-oriented architectures, such as UML4SOA [13] from the SENSORIA project [1], UPSS (UML Profile for Software Services) [11] adopted by IBM, and OMG's Service oriented architecture Modeling Language (SoaML) [15].

For the implementation and deployment of services, there are several alternatives. One of them is Apache Tuscany [19], which provides a service-oriented architecture infrastructure to develop and run applications in a service-oriented approach. It implements the Service Component Architecture (SCA), an assembly model developed by major vendors for composing heterogenous applications. EclipseSOA [8] is another interesting alternative, since it provides a runtime and tool integration platform that assists in all the required steps for the development of a SOA solution. Regarding the Enterprise Service Bus (ESB), there exist several open source ESB implementations, such as Mule [14].

As in the case of the processes, we will research on these tools in order to choose the one that better fulfils our needs.

4 Conclusions

New Future Internet applications will need support from trust and reputation services for their successful adoption. Whereas traditional security requirements have been provided by architecture middlewares and frameworks, trust and reputation have often been laid aside.

On the other hand, the service-oriented paradigm provides many advantages, including its focus on reusability. We propose the building of a service-oriented architecture, capable of delivering security services in general, and emphasizing the delivery of trust and reputation services in particular. This architecture must be flexible and reusable, constituting a framework that can be used by different applications that might require their services.

The main contributions that we expect to do in the field of Engineering Secure Software and Systems are in different levels. At a process level, we will research on the high-level development processes for building service-oriented solutions. At the architectural level, a research will be conducted regarding the modeling issues for service-based architectures. At a security level, the relationships between security services will be analyzed and made explicit, which in turn might help us to gain a better understanding of how trust is related to more traditional security services. Finally, from an overall perspective, we believe that the process of building a service-based security solution bridges a gap between the service engineering and security research communities.

References

1. SENSORIA - Software Engineering for Service-Oriented Overlay Computers. http://www.sensoria-ist.eu/, 2005–2010.
2. Selection and Documentation of the Two Major Application Case Studies. NESSoS Deliverable 11.2, October 2011.
3. Using TOGAF to Define and Govern Service-Oriented Architectures. Technical report, The Open Group, November 2011.
4. Cristian Aurel. *Service Oriented Security Architecture applied to Spatial Data Infrastructures*. PhD thesis, University of the Federal Armed Forces, Munich, January 2008.
5. Michael Bell. *Service-Oriented Modeling (SOA): Service Analysis, Design, and Architecture*. Wiley, 2008.
6. Philip Bianco, Grace A. Lewis, Paulo Merson, and Soumya Simanta. Architecting service-oriented systems. Technical report, Software Engineering Institute, August 2011.
7. Trung Dong Huynh. A Personalized Framework for Trust Assessment. *ACM Symposioum on Applied Computing - Trust, Reputation, Evidence and other Collaboration Know-how Track*, 2:1302–1307, December 2008.
8. Eclipse. EclipseSOA. http://www.eclipse.org/eclipsesoa/.
9. Ian Gorton. *Essential Software Architecture*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
10. Michael Hafner, Mukhtiar Memon, and Ruth Breu. SeAAS - A Reference Architecture for Security Services in SOA. *Journal of Universal Computer Science*, 15:1–21, December 2009.
11. Simon Johnston. UML 2.0 Profile for Software Services. Technical report, IBM, April 2005. http://www.ibm.com/developerworks/rational/library/05/419_soa/.
12. Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, March 2007.
13. Philip Mayer, Nora Koch, Andreas Schroeder, and Alexander Knapp. The UML4SOA Profile. Technical report, LMU Muenchen, 2010.
14. MuleSoft. Mule ESB. http://www.mulesoft.org/.
15. OMG. Service oriented architecture Modeling Language (SoaML). Technical report, OMG, April 2009.
16. Florian Siefert Gerrit Anders Theo Ungerer Wolfgang Reif Rolf Kiefhaber. The Trust-Enabling Middleware: Introduction and Application. pages 1–18, March 2011.
17. David Sprott and Lawrence Wilke. Ewerare-cbdi forum service-oriented architecture. http://msdn.microsoft.com/en-us/library/aa480021.aspx#aj1soa_topic3, January 2004.
18. Girish Suryanarayana, Mamadou H. Diallo, Justin R. Erenkrantz, and Richard N. Taylor. Architectural Support for Trust Models in Decentralized Applications. In *Proceeding of the 28th international conference*, pages 52–61, New York, USA, 2006. ACM Press.
19. The Apache Software Foundation. Apache Tuscany. http://tuscany.apache.org/home.html.
20. Ueli Wahli, Lee Ackerman, Alessandro Di Bari, Gregory Hodgkinson, Anthony Kesterton, Laura Olson, and Bertrand Portier. Building SOA Solutions Using the Rational SDP. Technical report, IBM, April 2007.

Access Control Policy Administration supporting User-defined Privacy Preferences

A Use-case in the context of Patient-centric Health-care

Thomas Trojer and Ruth Breu (Supervisor)

Institute of Computer Science, University of Innsbruck, Austria
thomas.trojer@uibk.ac.at

Abstract. The protection of medical records is understood to be an issue related to privacy and therefore closely bound to the patient her-/himself, playing a crucial role in networked electronic health-care. Awarding users to have control over personal data stored and processed by information systems is important as it allows a user to communicate individual privacy concerns. Still, users self-maintaining controls of access to their personal data poses challenges regarding its implementation. A major issue is that users are typically non-security experts and have only limited knowledge of the context domain. Regarding our use-case patients may not be fully familiar with all activities related to information processing e.g., during a medical treatment, therefore not able to properly decide on privacy and authorization measures. In our work we discuss the development of access control authoring tools to allow non-expert users to create, analyse and adjust personal privacy policies. We propose the integration of domain aspects into the development process of such tools. With extended knowledge about the domain the creation of policy rules can be bound to high-level activity descriptions and policy analysis can be performed in a domain-aware manner.

1 Motivation

Modern information systems are able to store, retrieve an process vast amounts of data. Further extended by networking capabilities and driven by the increased personal use of information technology a wide range of data can be collected and combined to form new processable content. The collection of data can be critical without means of regulating access to it when requests for provisioning or processing are made. With respect to the actual use-case, data can be considered as e.g., confidential according to its content or sensitive in terms of identifying individual persons.

In a common use-case scenario a person responsible for security matters, like an administrator, defines access control policies which constitute appropriate security measures for all protected resources. In the case of person-identifying information, such policies do not necessarily reflect the conception of the identified individual on how to access-protect these information. By declaring privacy as a right about *information self-determination*, e.g., within the *European Data*

Protection Directive[1] an individual user is awarded a distinguished role within privacy management processes. This can be interpreted as the required ability of a user to influence the definition of enforceable access control policies which constitute data privacy related to the user's personal conception.

1.1 Use-case: Patient-centric Electronic Health-care

We consider a use-case from the Austrian e-Health initiative which started in 2006 as a governmental workgroup. A central goal of this initiative is the establishment of a distributed shared electronic health-record for all citizens of Austria. It has been shown that a holistic medical history of a patient improves the health-care infrastructure from an economical perspective as well as from a viewpoint of effectiveness regarding medical treatments. Still, because of the high degree of sensitivity which is observed in most medical data, privacy is a concern of utmost importance to be tackled. In the context of this initiative we want to contribute methods with a strong focus on patient-centricity by establishing personal control over privacy-relevant health data.

2 Problem statement

A general problem question can be raised as follows: *How can a user, considered a non-security and non-domain expert, be supported during the declaration of access control policies in a way that she/he is aware about consequences to certain evaluation criteria, first and foremost personal privacy and e.g., the effectiveness of the information system.* Two potential user actions can be derived from this problem question. First, a user has to be provided with tools to create access control policies and second a currently active policy has to be visualized in a way that the user can understand how it influences the information flow of the system. Visualizing active policies is especially important to allow a user to reconsider the policies' appropriateness. Therefore a user is able to adjust a policy in a way so that it fits her/his personal conception of access control.

3 Contributions

We contribute a framework using domain characteristics to develop user interfaces for access control policy authoring. Further this framework includes a policy analysis component capable of providing users with feedback during their actions within the policy authoring process. A central step to be performed is therefore the modeling of domain entities and their relationships as well as the annotation of the domain model with attributes from the access control domain. In the context of our use-case we identify e.g., a *medical practitioner* or *pharmacist* as *subjects* of access control, whereas *medical records* or *referrals* are considered

[1] see Directive 95/46/EC, `http://ec.europa.eu/justice/policies/privacy/law/index_en.htm`

resources to be protected by access control. A basic API to work with domain entities and access control policy elements can be generated from that model.

Based on this model we designed a generic authoring process [10] that leads to the creation of access control policies. This process allows for policy creation and adaptation, which can be triggered by the user.

In this work we want to emphasize two research directions in order to reach our objective of providing a non-expert user with access control authoring tools to establish privacy policies. These are described in the following sections.

3.1 Scenario-based Access Control Policy Authoring

John Carroll [3] coined the term *scenarios* as stories about people and their activities, e.g., related to the tasks of their work. We propose *scenario-based authoring* as a method to create and further visualize access control policies in a usable way. This is important as users typically lack of knowledge about the underlying access control concepts and therefore have to be supported during the authoring process [12, 2].

The first step of our approach regards the elicitation of typical working activities of the domain. Only working activities which involve information processing in an arbitrary way are considered as they can be related to access control. In our context we tackled this step by performing a case-study about stakeholders and some of their activities in the domain of electronic health records in Austria [9]. Next the selected working activities have to be translated to our template language. A template consists of the attributes identifying and describing the working activity in natural language and further a set of access control rules which are written to the user policy once a template instance is executed. User control is established via user interface form input fields which represent domain and environment information (e.g. time, date, location or cardinalities) and are bound to variables used within the policy rules of the template. Fig. 1 shows a basic scenario from our electronic health-care use-case, namely, the selection of a family practitioner performed by a citizen stakeholder. In this example two inputs are provided, the name of the family practitioner and whether all documents (i.e. also all future ones) shall be accessible or only the ones currently stored about a patient. By executing a template instance one permit-rule is created allowing the selected practitioner (i.e. the subject of the access control target) to access patient health records. Similarly a query can be formed to visualize a selected family practitioner to the user by asking who is permitted to read all documents.

The template language including access control rules and queries are currently developed and described in a formal way. With this work we target the field of usable security.

3.2 Domain-based Access Control Policy Analysis

We see two situations where a user may be encouraged to reconsider her/his access control settings and to adjust them if necessary. First, if the representation

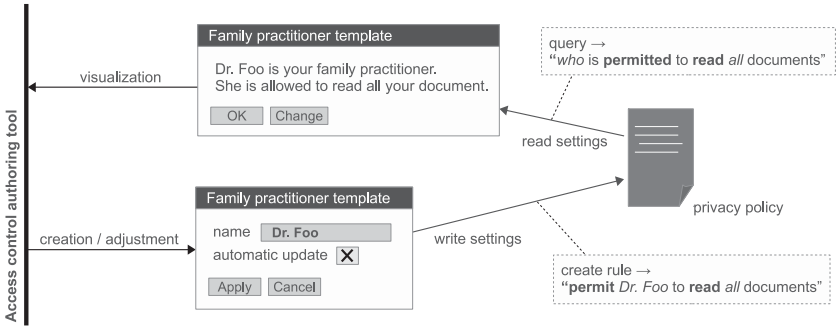

Fig. 1. Scenario-based policy authoring leading to privacy policies and visualization.

of currently active settings can be provided in a readable way the user is able to detect differences between these settings and her/his intended settings. Therefore we propose to use scenario-based policy templates (see Section 3.1) to increase understandability of a user policy.

Besides a user adjusting her/his policy based on a manual interpretation, policies may also carry conflicts or influence arbitrary system properties and the user in a negative way. Therefore we identify a second situation for adjusting a policy, namely triggered by feedback of a performed policy analysis. In general policy analysis is extensively discussed in literature (see e.g., [7, 8, 1]), but mainly based on conflict detection regarding the interplay of different policy rules. Work, as e.g., done by Michael LeMay [6] and Katie Fisler [4] consider a policy model together with the domain where policies are deployed on. Based on these works we propose the definition of high-level evaluation criteria which interact. These criteria can be attached to the policy authoring activity leading to a balancing act during access control configuration in order to satisfy best all evaluation criteria. In our previous work [9] we considered *privacy* and *information system effectiveness* as evaluation criteria to be balanced. There, based on a domain model and models for each evaluation criteria, domain-aware analysis rules integrating all evaluation aspects can be generated.

For our use-case we defined a trivial privacy model consisting of permissions and restrictions and an information system effectiveness model. This effectiveness model consists of personal relationships between subjects and needs-to-know relations between subjects and protected resources (see Fig. 2). Regarding privacy the lack of a permission can be interpreted as increased privacy. On the other hand the absence of one or both the personal relationship or the needs-to-know relation decreases the need of the information system to be effective towards these attributes. E.g. a family practitioner earns an associated personal relationship connecting her/him to the patient, further needs-to-know relations to all patient's data are established. Now, a patient restricting this practitioner from reading any data would obviously contribute to the her/his privacy, still the health-care information system would not effectively operate anymore. An

effectiveness warning with detailed information about its reasons is provided to the user, which in turn may react on it by adjusting her/his settings.

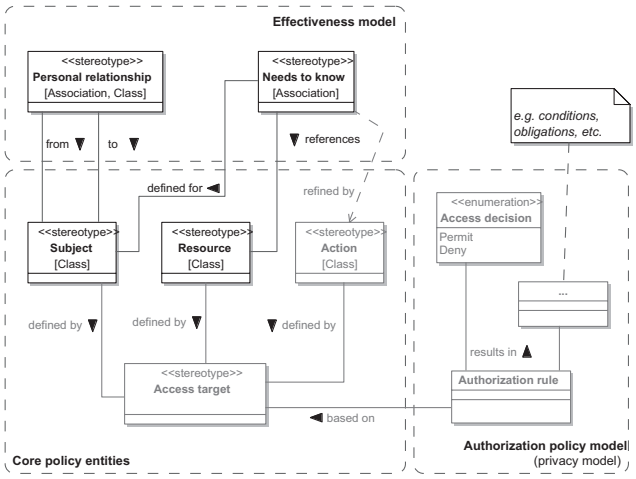

Fig. 2. Evaluation criteria *privay* and *effectiveness* applied to core policy entities and used for domain-aware policy analysis.

4 Research Plan

In an ongoing project with our industry partner *ITH-icoserve for healthcare technology*, a subsidiary company of Siemens and a local hospital provider, we are developing an access control policy authoring application based on a secured IHE-based infrastructure [2] for shared patient health-records.

Currently we have considered access control enforcement based on IHE XDS[2] and auxiliary profiles [5, 10], for which our industry partner is an implementer and tested for conformity and interoperability. Further a prototypical authoring portal application was developed [9]. In order to let the policy authoring reflect the actual domain, we employ a model-driven process which generates a policy API based on a domain and access control model [11]. Our approach for domain-aware policy analysis, which is based on balancing of evaluation criteria will also build upon the policy API. Evaluation criteria we currently consider is the correlation between permitted or restricted access, personal relationships between stakeholders and the importance to have certain data available to specific stakeholders. In future work we also want to study other criteria, e.g., the purpose-relatedness of permitted data accesses.

[2] see IHE IT-Infrastructure Technical Framework, `http://www.ihe.net/Technical_Framework/index.cfm#IT`

Generally we apply methods from design science to develop the aforementioned artifacts for patient-controlled access control. Usable methods for authoring and analysis of policies are our main focus. Further we will perform additional case studies to justify the application of these approaches within our use-case. As human interaction with the authoring application is a central part of this work, therefore we will also conduct a usability study to evaluate the usefulness of working scenarios and templates to maintain access control policies. A fully features authoring portal application is planned to be integrated into a health information system built by our industry partner and deployed to our regional health-care infrastructure. This will consist of templates for adapting authorization policies as well as policy analysis to inform a citizen about the consequences of certain access control settings. Finally the deployed system has to be evaluated regarding its performance and user acceptance.

References

1. E. Bertino, B. Catania, E. Ferrari, and P. Perlasca. A logical framework for reasoning about access control models. In *Proceedings of the sixth ACM symposium on Access control models and technologies*, SACMAT '01, 2001.
2. S. Brostoff, M. A. Sasse, D. Chadwick, J. Cunningham, U. Mbanaso, and S. Otenko. R-What? Development of a Role-Based Access Control (RBAC) Policy-Writing Tool for e-Scientists. *Software: Practice and Experience*, 35(9):835–856, July 2005.
3. J. Carroll. Five reasons for scenario-based design. *Interacting with Computers*, 13(1):43 – 60, 2000.
4. K. Fisler and S. Krishnamurthi. A model of triangulating environments for policy authoring. In *Proceeding of the 15th ACM symposium on Access control models and technologies*, SACMAT '10, 2010.
5. B. Katt, T. Trojer, R. Breu, T. Schabetsberger, and F. Wozak. Meeting ehr security requirements: Seaas approach. In *EFMI STC 2010. Accepted*, June 2010.
6. M. LeMay, O. Fatemieh, and C. A. Gunter. PolicyMorph: interactive policy transformations for a logical attribute-based access control framework. In *Proceedings of the 12th ACM symposium on Access control models and technologies*, SACMAT '07, 2007.
7. E. Lupu and M. Sloman. Conflicts in Policy-based Distributed Systems Management. *IEEE Transactions on Software Engineering*, 25, 1999.
8. J. D. Moffett and M. S. Sloman. Policy conflict analysis in distributed system management, 1993.
9. T. Trojer, B. Katt, T. Schabetsberger, R. Breu, and R. Mair. Considering privacy and effectiveness of authorization policies for shared electronic health records. In *ACM IHI 2012 (in press)*, 2012.
10. T. Trojer, B. Katt, T. Schabetsberger, R. Mair, and R. Breu. The Process of Policy Authoring of Patient-controlled Privacy Preferences. In *eHealth 2011*.
11. T. Trojer, B. Katt, F. Wozak, and T. Schabetsberger. An Authoring Framework for Security Policies: A Use-case within the Healthcare Domain. In *eHealth 2010*, 2010.
12. T. Whalen, D. Smetters, and E. F. Churchill. User experiences with sharing and access control. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06, pages 1517–1522, New York, NY, USA, 2006. ACM.

Federated authorization for SaaS applications

Maarten Decat, Bert Lagaisse, Wouter Joosen

IBBT-DistriNet, KU Leuven, 3001 Leuven, Belgium

Abstract. With Software-as-a-Service (SaaS), a centrally hosted web-based application is offered to a large number of customer organizations called tenants, each using multiple applications. The tenant and provider each work in their own authoritative and administrative domain, leading to a federated architecture and raising the bar for security and access control. Access control with SaaS applications is about protecting the tenant's data at the provider's side using the tenant's policies and user information. In current practice however, all access control policies are evaluated at the provider's side, distributing and fragmenting the tenant's policies over the multiple applications it uses. Moreover, all necessary user information now has to be shared with the provider, resulting in the disclosure of confidential tenant data. Therefore, we propose the concept of *federated authorization*, a combination of externalized authorization and federated access control techniques whereby the tenant's access control policies are evaluated at the tenant's side using local data.

1 Introduction

Software-as-a-Service (SaaS) applications are a part of cloud computing in which centrally hosted web-based applications are offered to a large number of tenants (customer organizations each representing multiple end-users), each using multiple applications. From the point of view of the tenant, cloud computing and SaaS are a form of outsourcing using a pay-per-use billing scheme. From the point of view of the provider, SaaS is the next step in Application Service Provider (ASP) evolution, trying to cut operational costs by sharing resources of the application and offering it on a larger scale.
Originally, SaaS applications where mainly used by small and medium enterprises for non-core business applications. Typical examples are Google Apps (an office suite, e-mail, calendar etc.) and Salesforce (CRM). Recently however, large enterprises are increasingly starting to use SaaS for core-business applications. An example of this is a home patient monitoring system offered to health institutions like hospitals (the tenants of the application). With regards to the typical examples, these applications pose new challenges, such as the increased importance of security. While cloud computing and SaaS can provide large economical advantages, security is the major hurdle for cloud adoption [8,6,5].
SaaS requires application-level security, of which access control is an important part. From a high-level point of view, access control limits the actions a subject (e.g., a user) can take on an object in the system (e.g., a file) applying rules

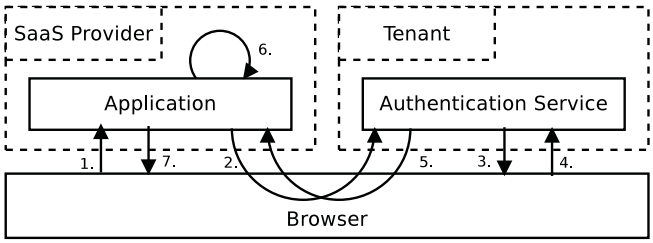

Fig. 1. Federated authentication, SAML flow [1]: the user requests a certain resource using his browser (step 1), the SaaS application redirects the user to its home domain (step 2), the user authenticates himself locally (step 3 and 4), the user is redirected back to the application together with its user information (step 5), the application uses this information for access control (step 6) and returns the resource if allowed (step 7).

defined in access control policies. The process of access control is generally divided into authentication, authorization and audit. Authentication confirms the stated identity of a subject, authorization confirms the subject is allowed to do the desired action on the object and audit offers information about the actions done in the system.
This paper focuses on authentication and authorization. Section 2 defines the problems we observe with current state of the art for authorization in SaaS applications. Section 3 sketches our proposed solution and specific objectives. Section 4 lists the challenges we predict in this research. Section 5 describes our research methodology and section 6 shows our main contributions.

2 Problem statement

SaaS applications are a form of outsourcing: the application remotely hosts and processes data actually belonging to the tenant. Therefore, access control in SaaS applications is mainly about protecting the tenant's data located at the provider's side. The tenant controls the administration and information about the users of the application[1] (e.g., employees of the tenant) and the access control rules and policies.

A first problem with access control for remote applications is *administrative scalability*. From the tenant's point of view, every application has to be provided with the necessary user information. Therefore, *federated authentication* techniques such as WS-Federation [3], OpenId [2] and SAML [1] have been developed. These techniques allow users to be authenticated in their home domain, after which the needed user information is securely exchanged with the application (see fig. 1). This centralizes user management with the tenant and offers control over the shared information and the authentication means used.

[1] There are more complex scenario's in which former unknown users access the application (e.g., ad-hoc federations). Here we purposely limit our vision and exclude this for now.

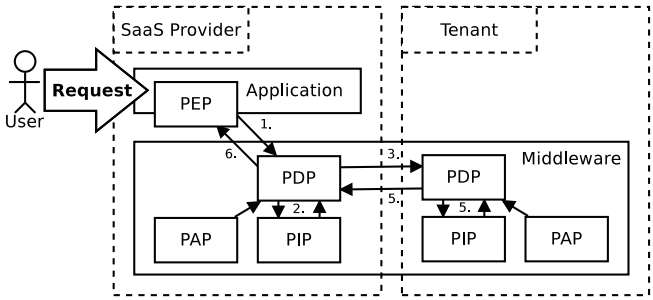

Fig. 2. Externalized authorization

With the evolution towards core-business SaaS applications, more control over the application and the data it uses is required, raising the need for authorization. While federated authentication techniques allow tenants to centralize user management, similar techniques for authorization currently do not exist and all access control policies are evaluated at the provider's side. Because of this, access control policies are distributed and fragmented over the multiple SaaS applications the tenant uses. This increases the administrative load, eventually leading to inconsistencies and security holes.
A second problem with this approach, is the necessary *disclosure of confidential tenant information*. Because the access control policy is evaluated at the provider's side, all necessary information has to be shared with the provider. While the tenant trusts the provider with the information in the application itself, it is likely that more information is needed for authorization. For example, in the patient monitoring system, the list of all patients the physician is currently treating or the fact that the patient is currently being treated by an oncologist is confidential information. Moreover, it can be the case the tenant does not want to share the policy itself, for example how the hospital handles its competitors when requesting access to its data.

3 Proposed solution

In this section we present an initial solution we envision to solve the two problems described above: their limited administrative scalability and the necessary disclosure of confidential information. The solution we envision consists of externalizing policy evaluation from the SaaS application to the tenant. With each request, the SaaS application would ask the tenant for an access control decision, after which the tenant himself evaluates the necessary policies and returns his decision. This way, tenant policies remain centralized and confidential information does not have to be shared with the application provider. Since this achieves for authorization what federated authentication did for authentication, we call this *federated authorization*.

The objective of this research is to incept and evaluate support for federated authorization in security middleware. A high-level technical overview of how we

currently envision this middleware is given in fig. 2 using XACML [11] terminology for the components. When a user sends a request to the SaaS application to perform an action, the Policy Enforcement Point (PEP) is called from the application in order to determine whether the user is allowed to do so. It therefore calls the local Policy Decision Point (PDP), which will evaluate tenant policies loaded from the Policy Administration Point (PAP) using information stored in the Policy Information Point (PIP). The provider's PDP also forwards the request to the tenant PDP, which in turn evaluates the tenant's policies using local data and returns its decision. The requested action is only allowed in case both parties allow it.
In order to achieve this objective, both the policy language and the execution environment will have to be extended. The policy has to support the necessary constructs in order to declare the needed properties of the federated authorization. Future research will have to show which constructs are minimally needed. For the execution environment, the communication between the provider and the tenant will most likely be the crucial part, since the information about the request, the subject, the object and the system are now distributed over the two parties.

4 Challenges

Federated authorization as described above builds upon and extends externalized authorization, a concept that has been researched in the past (e.g., [9]). Moreover, the XACML access control standard already defines a profile for integration with SAML for transport of both user information and authorization decisions [10]. However, no research about the practical feasibility of combining externalized authorization with federation into federated authorization is present to the best of our knowledge. An important aspect of this are the security tradeoffs implicitly made in federated authorization. For example, externalizing access control to the tenant increases administrative scalability, but also makes the system more susceptible to denial of service attacks and traffic analysis.
One major challenge we predict is the performance and performance scalability of the security middleware. This challenge is increased by taking into account concurrency and distributed data updates. Related research has led to techniques for improving access control performance (e.g., Wei [14] proposes decision recycling and inference, Brucker and Petritsch [4] focus on the retrieval of the necessary user information), some of which can also be applied for scalability. However, SaaS applications frequently use aggressive scalability techniques [12] and their applicability on access control still has to be investigated.

5 Research methodology

In this research, a case-study driven methodology is used. Two major case studies are used: (i) a home patient monitoring system and (ii) an electronic document processing system. The security analyses of both case studies offer require-

ments and policies to be applied in our prototypes. The former offers insights in the stringent legal requirements concerning medical data and the complex, fine-grained access control policies in e-health, the latter offers a tenant hierarchy, a concept which should be supported by federated authorization.
Currently, this research is in its early phase. Involvement in the case studies stated above has shown the need for solving this problem, after which a literature study of existing authorization techniques was made. As the next step in this research, our concept of federated authorization will be refined into a reference architecture. In order to build upon existing work, we will base this on widely-used standards such as XACML [11] as much as possible. This reference architecture will then be instantiated into a proof of concept using the case studies in order to evaluate the feasibility of federated authorization. The next step will be to apply and empirically evaluate performance and scalability techniques on this using metrics such as the impact of the complexity of policies (e.g., the number of subject and object attributes needed) on the response time of the whole.

6 Main contributions

Next to SaaS applications, federated access control has been researched in other domains as well, such as web applications (e.g., OpenId [2]), grid computing (e.g., PERMIS [7]) and web services (e.g., WS-Federation [3]). These domains offer relevant techniques, but their purpose differs from SaaS. For example, access control in grid computing is mainly about access to on-premise resources from other domains. Moreover, the described techniques still limit themselves to local policy enforcement based on federated authentication.
Some literature has been published about access control for multi-party systems. Zhang et al. [15] describe a framework for usage control in collaborative systems, taking into account access control data updates. Stihler et al. [13] also describe an architecture which enables the involved parties to declare access control policies on the application. However, both still employ provider-side policy enforcement, resulting into the problems described in section 2.
Existing technologies for access control also do not support federated authorization. Products such as IBM Tivoli Access Manager allow to centrally enforce access control policies on multiple applications. Moreover, by acting as an *XML gateway* these policies can be applied on external applications. This set-up is however limited to the level of messages, thereby limiting the complexity of supported policies. Moreover, this limits the mobility of the users of the external application, making it inapplicable for SaaS applications.

7 Conclusions

This paper presented a new approach to access control for SaaS applications called *federated authorization*. In this approach, policy evaluation is externalized from the SaaS application to the tenant, keeping tenant policies centralized

and its data confidential. A major challenge we predict is the performance and performance scalability of the security middleware. Using a case-study driven methodology, we will evaluate the feasibility of federated authorization for SaaS applications.

Acknowledgements This research is partially funded by the Interuniversity Attraction Poles Programme Belgian State, by the Belgian Science Policy, by the Research Fund KU Leuven, by the EU FP7 project NESSoS and by the Agency for Innovation by Science and Technology in Flanders (IWT).

References

1. Security Assertion Markup Language (SAML) v2.0 (March 2005), http://www.oasis-open.org/standards#samlv2.0
2. OpenID Authentication 2.0 - Final (December 2007), http://openid.net/specs/openid-authentication-2_0.html
3. Bajaj, S., Della-Libera, G., Dixon, B., Dusche, M., Hondo, M., Hur, M., Kaler, C., Lockhart, H., Maruyama, H., Nadalin, A., et al.: Web Services Federation Language (WS-Federation). http://specs.xmlsoap.org/ws/2006/12/federation (December 2006)
4. Brucker, A., Petritsch, H.: Idea: Efficient Evaluation of Access Control Constraints. Engineering Secure Software and Systems pp. 157–165 (2010)
5. Brunette, G., Mogull, R.: Security guidance for critical areas of focus in cloud computing v2.1. Tech. rep., Cloud Security Alliance (december 2009)
6. Catteddu, D., Hogben, G.: Cloud Computing: benefits, risks and recommendations for information security. Tech. rep., ENISA (november 2009)
7. Chadwick, D., Otenko, A.: The PERMIS X. 509 role based privilege management infrastructure. Future Generation Computer Systems 19(2), 277–289 (2003)
8. Dean, D., Saleh, T.: Capturing the Value of Cloud Computing. Tech. rep., Boston Consultancy Group (november 2009)
9. Karjoth, G.: Access control with ibm tivoli access manager. ACM Transactions on Information and System Security (TISSEC) 6(2), 232–257 (2003)
10. Lockhart, H., Parducci, B., Rissanen, E., Lockhart, H.: SAML 2.0 Profile of XACML, Version 2.0
11. Parducci, Bill and Lockhart, Hal and Rissanen, Erik: eXtensible Access Control Markup Language (XACML). http://www.oasis-open.org/committees/xacml (August 2010)
12. Shoup, R.: Scalability Best Practices: Lessons from eBay (May 2008), http://www.infoq.com/articles/ebay-scalability-best-practices
13. Stihler, M., Santin, A., Calsavara, A., Marcon, A.: Distributed usage control architecture for business coalitions. In: Communications, 2009. ICC'09. IEEE International Conference on. pp. 1–6. IEEE (2009)
14. Wei, Q.: Towards improving the availability and performance of enterprise authorization systems. Ph.D. thesis, University of British Columbia (2009)
15. Zhang, X., Nakae, M., Covington, M., Sandhu, R.: Toward a usage-based security framework for collaborative computing systems. ACM Transactions on Information and System Security (TISSEC) 11(1), 1–36 (2008)

Modeling Social Networking Privacy

Carolina Dania*

IMDEA Software Institute, Madrid, Spain
Universidad Complutense de Madrid, Spain
carolina.dania@imdea.org

Abstract. Privacy-related issues are becoming a serious concern among users of social networks. There are at least three reasons that justify this growing concern: social networking privacy policies are hardly trivial; they live in a constant state of flux; and they are only informally and partially described by the social networking sites.
To improve this current state of affairs, we propose SecureUML as a formal language to model social networking privacy, and we set ourselves the goal of modeling, as a case study, Facebook privacy policy. Based on our formal model, we envision a new generation of tools that will provide Facebook users with more information about the privacy of their posts and about the associated privacy-related risks.

1 Motivation

Many people in our society rightly consider themselves as "internet natives": when they need information, they naturally open a browser and search for it; when they want to share information, they naturally post it on a social network. A few figures about Facebook, the leader among social networking sites, exemplify our point: Facebook has more than 800 million users, of which, about 50% log on to their accounts every day; more to the point, Facebook users upload, on average, 250 million photos per day [5].

Not surprisingly, privacy-related issues are a growing concern among users of social networking sites [7, 1, 14, 15] and, consequently, among their developers. Last November, Facebook's founder and CEO, Mark Zuckerberg, wrote in his blog [16] "I also understand that many people are just naturally skeptical of what it means for hundreds of millions of people to share so much personal information online, especially using any one service. Even if our record on privacy were perfect, I think many people would still rightfully question how their information was protected." Then, recognizing an increasing criticism over Facebook privacy policy, Zuckerberg announced: "we're making a clear and formal long-term commitment to do the things we've always tried to do and planned to keeping doing —giving you tools to control who can see your information and then making sure only those people you intend can see it."

To Facebook's credit, over the past 2 years, its users have been equipped with new tools and resources which are designed to give them more control over

* This work has been partially supported by the EU-NoE project NESSoS, 256980.

their Facebook experience, including: an easier way to select your audience when making a new post; inline privacy control on all your existing posts; the ability to review tags made by others before they appear on your profile; a tool to view your profile as someone else would see it; many more privacy education resources.

Despite all these efforts, many users are still concerned about how to maintain their privacy or —in Zuckerberg's words— "rightfully questions how their information was protected". There are at least three reasons for this:

- The Facebook privacy policy is hardly trivial to understand: for example, when tagging policies and privacy settings conflict to each other.
- The Facebook privacy policy has been in a constant state of flux over the past few years [12], and it is prompted to change again in the near future.
- The Facebook privacy policy is only informally and partially described in a collection of "privacy education resources", which cannot provide a coherent and complete account of the policy.

As a consequence, even advanced Facebook users may find difficult to understand, for example, the actual visibility of a post. To illustrate our point, recall the tagging policy explained in [6]:

> "*When I tag someone in a photo or post, who can see it?* When you tag someone, it may be visible to: 1. The audience you selected for your post. 2. Friends of the person you tagged (if the audience is set to "Friends" or more). (...) *When someone adds a tag to a photo or post I shared, who can see it?* When someone adds a tag to something you shared, it's visible to: 1. The audience you chose for the post or photo. 2. The person tagged in the post, and their friends (if the audience is set to "Friends" or more)."

Now, suppose that Bob, Alice, Ted, and Peter have Facebook profiles: Bob is friend of Alice and Ted; Ted is friend of Peter; Ted is not a friend of Alice; and Peter is not friend of Alice or Bob. Assume also that Alice has set to "Friends" the default audience for posts of friends in her wall. Consider the following scenarios:

Scenario #1 Alice posts a photo of herself, Bob and Ted in her wall, and set its audience to "Friends". Then, Bob tags Ted in this photo. *Question: Can Peter see this photo in Alice's wall?* (The answer is Yes.).

Scenario #2 Bob posts a photo of himself, Ted and Alice in Alice's wall. Then, Bob tags Ted in this photo *Question: Can Peter see this photo in Alice's wall?* (The answer is No. Why?).

2 Research Project

Objectives. We set ourselves two goals: first, to provide a formal account of the Facebook privacy policy (as complete as possible); and second, to design methods (based on this formal account) for reasoning about sharing and privacy in Facebook.

Potential impact. We envision at least three new Facebook privacy tools that can use our results as a solid, rigorous foundation: first, a tool for checking whether a person can see a post (currently, this tool is only available for the owner of the wall where the post is posted, but not for the creator of the post); second, a tool for assessing the risk of a post becoming visible for a person; and third, a tool for assessing the impact, on the visibility of a post, of a default privacy policy change. We also expect our methodology to be applicable to other social networking site, like Google+, opening the path for a formal comparison between privacy policies of different social networking sites.

Context. This project is being conducted at IMDEA Software (`http://software.imdea.org`) Modeling Lab, under the co-supervision of Manuel Clavel and Marina Egea. Manuel Clavel is Associate Researcher at IMDEA Software and Professor at Universidad Complutense de Madrid. Dr. Marina Egea is Consultant & Research Project Manager at Atos S.A., Spain. We are developing this research within the European Network of Excellence for Engineering Secure Software and Systems.

Methodology. As discussed in [13], when modeling social networking privacy it is crucial to use a language able to formalize *fine-grained* access control policies: in other words, a role-based access control language, as proposed in [9], will only do part of the job. There are differents options for this, but not so many when having a formal semantics becomes a hard requirement. For example, XACML [10], which can be considered the standard choice for describing privacy policies, lacks of a formal semantics.

To provide a formal account of the Facebook privacy policy, we use SecureUML [3]. SecureUML is a formal language for modeling role-based access control. It provides a rich language for expressing both static and dynamic access control policies, the latter being policies that depend on the run-time satisfaction of authorization constraints. Based on our preliminary results, we believe that SecureUML is up for the task we have set to ourselves for the following reasons:

1. Facebook ultimately decides about the visibility of a post based on the settings chosen by the owner of the wall and on the relationships (if any) that link the visitor of the wall, the owner of the wall, the creator of the post, and the creators and targets of the tags (if any) added to the post. Interestingly, when only real users are considered (i.e., no Facebook-enhanced games, applications, websites, or advertisers) the *purpose* of the visitor (and, similarly, for the creator of the post or of the tags) play no role in Facebook decisions; neither assigns the Facebook privacy policy any *obligation* to the visitor.
2. Facebook's profiles, walls, posts, photos, tags, and so on, can be naturally modeled in SecureUML as *entities*, with the expected relationships between them: the owner of a wall, the creator of a post, the wall where a post is posted, the post where a tag is added, and so on. In particular, privacy settings can be modeled as *attributes* of the entities 'profile' and 'post' while the relationship of friendship can be modeled as a *self-association* of the entity 'profile'.

3. Facebook's policy constraints (like 'a user can read a post if he or she is a friend of the owner of the wall where the post is posted') are naturally modeled in SecureUML using OCL [11]. OCL is a strongly typed, declarative language, specifically designed for querying scenarios consisting of entities (with attributes) and associations between them. In particular, using OCL, we can (the list is by no means exhaustive):
   - refer to the value, in a data element, of any of the attributes specified in the data model.
   - refer to all the data elements which are linked to a data element through any of the associations specified in the data model.
   - perform standard operations on booleans (negation, conjunction, disjunction, implication, etc.).
   - perform equalities/inequalites between (collection of) data elements of the same type.
   - perform standard operations on collections (union, intersection, emptiness, inclusion, exclusion, insertion, deletion, etc.).

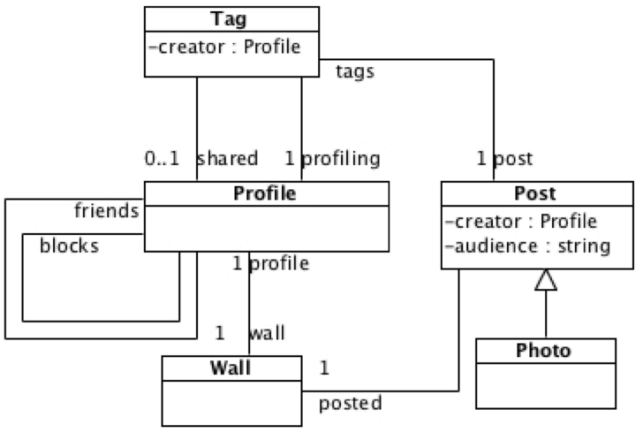

Fig. 1. Data model for Facebook posts and tags (partial).

To illustrate our methodology, we show in Figure 1 a basic data model that (partially) represents Facebook posts and tags. Using the entities, attributes, and associations contained in this data model, we show in Figure 2 how the following clauses —about when a visitor (`@caller`) can read a post (`@post`)— are formalized in OCL:

- anybody can read any post that is posted in his or her wall, independently of its creator.
- anybody can read any post that is posted in a wall when he or she is a friend of the owner of the wall and the audience selected is 'Friends'.

```
@caller=@post.posted.profile
or (@post.audience='Friends' and @post.posted.profile.friends->includes(@caller))
or (@post.posted.profile.blocks->excludes(@caller) and
        (@post.audience='Public'
         or @post.tags.profiling->includes(@caller)
         or (@post.audience= 'Friends' and @post.creator=@post.posted.profile
              and @post.tags.profiling.friends->includes(@caller))
)) or ...
```

Fig. 2. Authorization constraints for reading a Facebook post (partial).

- anybody can ready any post that is posted in a wall when the audience selected is 'Public', unless he or she is blocked by the owner of the wall.
- anybody can read any post that is posted in a wall when he or she is tagged in this post, independently of the audience selected, unless he or she is blocked by the owner of the wall.
- anybody can read any post that is posted in a wall, when the audience selected is 'Friends', he or she is a friend of somebody tagged on the post, he or she is not blocked by the owner of the wall, and the owner of the post happens to be the creator of the post.

On the other hand, with respect to our second goal (namely, reasoning), SecureUML has a well-defined semantics that supports the formal analysis of its models. In particular, for a certain type of analysis, we can automatically analyze SecureUML models using the metamodel-based approach described in [2]. For more general analysis, we can use theorem-proving tools (including SMT solvers), thanks to the mapping from OCL to first-order logic introduced in [4].

3 Research Plan

Our first task is to gather as much information as possible about the Facebook privacy policy. We would like to assume that the information available at [6] is correct and complete. Unfortunately, our initial results show that this is not always the case. Our second task is then to design "experiments" on precooked Facebook scenarios for testing that our understanding of the Facebook privacy policy corresponds to reality. Eventually, these "experiments" should also help us to monitor and report changes in the Facebook privacy policy. We will also look very closely at the results of the thorough and detailed audit [8] of Facebook's practices and policies by the Office of the Irish Data Protection Commissioner.

According to the gathered information, we will decide how to proceed. We envision separated tasks for modeling each of the basic actions on Facebook: select audience, switch reviews on/off, read a post, write a post, remove a post, add a tag, remove a tag, approve a tag, add a friend, remove a friend, block a user, and so on. For each of these actions, we plan to model also their pre- and post-conditions using OCL. We are aware that we may have to exclude from our

project the privacy policies that apply to advertisers and/or so-called Facebook-enhanced games, applications, and websites, unless we obtain this information directly from the company. Finally, we plan to design methods (based on the different types of analysis for SecureUML models that we mentioned before) for reasoning about sharing and privacy in Facebook (e.g., audience evaluation, risk and change impact assessment), although the actual implementation of these methods may have to be carried out in other research projects.

References

1. A. Acquisti and R. Gross. Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook. In G. Danezis and P. Golle, editors, *Privacy Enhancing Technologies*, volume 4258 of *LNCS*, pages 36–58. Springer, 2006.
2. D. Basin, M. Clavel, J. Doser, and M. Egea. Automated analysis of security-design models. *Information and Software Technology*, 51(5):815–831, 2009.
3. D. Basin, J. Doser, and T. Lodderstedt. Model driven security: From UML models to access control infrastructures. *ACM TOSEM*, 15(1):39–91, 2006.
4. M. Clavel, M. Egea, and M. A. G. de Dios. Checking unsatisfiability for OCL constraints. *Electronic Communications of the EASST*, 24, 2009.
5. Facebook. `http://www.facebook.com/press/info.php?statistics`.
6. Facebook. Facebook Help Center. `http://www.facebook.com/help`.
7. R. Gross, A. Acquisti, and H. J. Heinz. Information Rvelation and Privacy in Online Social Networks. In V. Atluri, S. D. C. di Vimercati, and R. Dingledine, editors, *WPES*, pages 71–80. ACM, 2005.
8. Irish Data Protection Commissioner. Facebook Ireland Ltd. Report of Audit, December 2011.
9. J. Li, Y. Tang, C. Mao, H. Lai, and J. Zhu. Role based access control for social network sites. In *Pervasive Computing (JCPC), 2009 Joint Conferences on*, pages 389 –394, dec. 2009.
10. OASIS. eXtensible Access Control Markup Language (XACML), 2010. `http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf`.
11. Object Management Group. *Object Constraint Language specification Version 2.2*, Feb. 2010. OMG document available at `http://www.omg.org/spec/OCL/2.2`.
12. N. O'Neill. Infographic: The History of Facebooks Default Privacy Settings. `http://www.allfacebook.com/`.
13. A. Simpson. On the Need for User-Defined Fine-Grained Access Control Policies for Social Networking Applications. In *Proceedings of the workshop on SOSOC*, pages 1:1–1:8, New York, NY, USA, 2008. ACM.
14. A. Young and A. Quan-Haase. Information Revelation and Internet Privacy Concerns on Social Network Sites: A Case Study of Facebook. In *Proceedings of the international conference on C&T*, pages 265–274, New York, NY, USA, 2009. ACM.
15. E. Zheleva and L. Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, editors, *WWW*, pages 531–540. ACM, 2009.
16. M. Zuckerberg. Our Commitment to the Facebook Community. The Facebook Blog, Nov. 2011. `https://blog.facebook.com`.

Patterns- and Security-Requirements-Engineering-based Support for Development and Documentation of Security Standard Compliant ICT Systems *

Kristian Beckers and Maritta Heisel (PhD Supervisor)

paluno - The Ruhr Institute for Software Technology University of Duisburg-Essen
{firstname.lastname}@paluno.uni-due.de

Abstract. Aligning an ICT system with a security standard is a challenging task, because of the sparse support for development and documentation that these standards provide.
We create patterns for the elements of trustworthiness: security, risk management, privacy, and law. The instantiations of these patterns are used to support the development and documentation of ICT systems according to security standards. In addition, we define relations between security standards and security requirements engineering approaches.

Key words: security standards, requirements engineering, security, patterns

1 Motivation and Background

Security is a system property of ICT systems [1, 2] and an acceptable security level has to be achieved for the entire system. Security standards exist that provide relevant methods for achieving this goal. However, aligning ICT systems with security standards is difficult, because the standards provide only sparse support for system development and documentation. For example, assembling an *information security management system (ISMS)* according to the ISO 27001 requires a *scope and boundaries* description among its initial steps. The required input is to consider "characteristics of the business, the organization, its location, assets and technology"[3, p. 4].

Security requirements engineering (SRE) methods, on the other hand, provide structured elicitation and analysis of security requirements. This structured elicitation and analysis of security requirements of SRE methods is useful for numerous security engineering contexts. Therefore, we propose to use SRE methods to support security engineers in the development and documentation of trustworthy ICT systems that are compliant to security standards.

This thesis is inspired by the work of Gamma et. al [4], which manages comprehensible to describe design problems and solutions in a fairly easy way. We

* This research was partially supported by the EU project Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS, ICT-2009.1.4 Trustworthy ICT, Grant No. 256980).

aim to accomplish the same for design and documentation problems of trustworthy ICT systems. Security engineering "requires cross-disciplinary expertise" [5, p. 3]. Patterns provide the means to collect this expertise and instantiate it to a given security engineering problem. We define trustworthiness as a combination of security, risk management, privacy and compliance attributes. All of these attributes are also required by security standards, e.g., ISO 27001. Hence, we restrict patterns in this work to security, law, privacy, and risk management patterns.

The outcome of this analysis answers the research question, if and to what extent patterns and SRE approaches can support the development of a security standard compliant ICT system. Moreover, it answers the question in what way patterns and SRE methods provide the required documentation for a security standard compliant ICT system and how existing pattern-based and SRE documentation can be re-used for an aforementioned system.

2 Previous Work

ICT systems keep increasing their functionality and distribution in recent years. Unfortunately this increase in complexity of ICT systems leads also to an increase in security problems for instance in cloud computing systems (or short clouds) [6].

We developed a pattern-based approach to support the context establishment and asset identification in the scope of cloud computing systems for the ISO 27005 [7] standard [8]. Our work shows a cloud system analysis pattern and different kinds of stakeholder templates serve to understand and describe a given cloud development problem. We illustrated our support using an online banking cloud scenario, presented in in Fig. 1. Our *cloud system analysis pattern* in Fig. 1 that provides a conceptual view on cloud computing systems and serves to systematically analyse stakeholders and requirements. The notation used to specify the pattern is based on UML[1] notation, i.e. the stick figures represent roles, the boxes represent concepts orientates of the real world, the named lines represent relations (associations) equipped with cardinalities, the unfilled diamond represents a "part-of" relation, and the unfilled triangles represent inheritance.

A *Cloud* is embedded into an environment consisting of two parts, namely the *Direct System Environment* and the *Indirect System Environment*. The *Direct System Environment* contains stakeholders and other systems that directly interact with the *Cloud*, i.e. they are connected by associations. Moreover, associations between stakeholders in the *Direct* and *Indirect System Environment* exist, but not between stakeholders in the *Indirect System Environment* and the cloud. Typically, the *Indirect System Environment* is a significant source for compliance and privacy requirements.

The *Cloud Provider* owns a *Pool* consisting of *Resources*, which are divided into *Hardware* and *Software* resources. The provider offers its resources as *Services*, i.e. *IaaS*, *PaaS*, or *SaaS*. The boxes *Pool* and *Service* in Fig. 1 are hatched,

[1] Unified Modeling Language: `http://www.omg.org/spec/UML/2.3/`

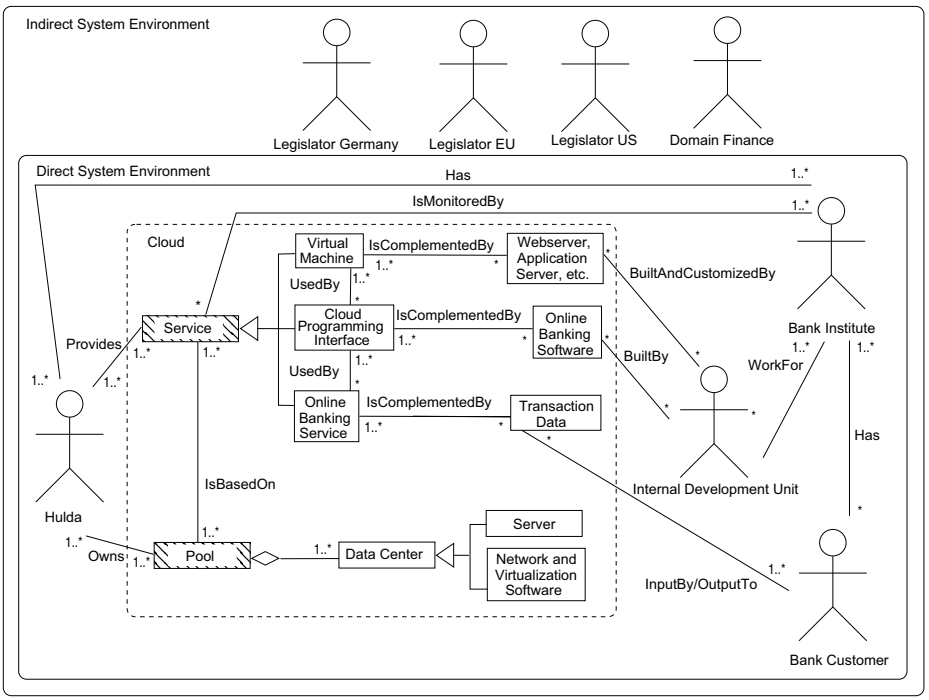

Fig. 1: Cloud System Analysis Pattern

because it is not necessary to instantiate them. Instead, the specialised cloud services such as *IaaS*, *PaaS*, and *SaaS* and specialised *Resources* are instantiated. The *Cloud Developer* represents a software developer assigned by the *Cloud Customer*. The developer prepares and maintains an *IaaS* or *PaaS* offer. The *IaaS* offer is a virtualised hardware, in some cases equipped with a basic operating system. The *Cloud Developer* deploys a set of software named *Cloud Software Stack* (e.g. web servers, applications, databases) into the *IaaS* in order to offer the functionality required to build a *PaaS*. In our pattern *PaaS* consists of an *IaaS*, a *Cloud Software Stack* and a *cloud programming interface (CPI)*, which we subsume as *Software Product*. The *Cloud Customer* hires a *Cloud Developer* to prepare and create *SaaS* offers based on the CPI, finally used by the *End Customers*. *SaaS* processes and stores *Data* in- and output from the *End Customers*. The *Cloud Provider*, *Cloud Customer*, *Cloud Developer*, and *End Customer* are part of the *Direct System Environment*. Hence, we categorise them as *direct stakeholders*. The *Legislator* and the *Domain* (and possibly other stakeholders) are part of the *Indirect System Environment*. Therefore, we categorize them as *indirect stakeholders*.

The cloud system analysis pattern instance in Fig. 1 helps, e.g., identifying assets by considering the instantiated boxes and the associations between the direct stakeholders and the cloud. The associations indicate the flow of information into and out of the cloud and therefore helps to analyze the information

assets processed and stored in the cloud. Furthermore, the associations help to find out about the asset owner, as the standard requires.

Identifying relevant compliance regulations for a software system and aligning it to be compliant is a challenging task. Hence, we already developed a pattern-based method for Identifying and analyzing laws [9]. The method makes use of different kinds of patterns, which help to systematically elicit relevant laws.

We also analyzed the ISO 27001 standard to determine what techniques and documentation are necessary and instrumental to develop and document systems according to this standard [10]. Based on these insights, we inspected a number of current SRE approaches to evaluate whether and to what extent these approaches support ISO 27001 system development and documentation. We re-use a conceptual framework (CF) [11] originally developed for comparing SRE methods to relate important terms, techniques, and documentation artifacts of the security requirements engineering methods to the ISO 27001.

3 Future Work

In the future we will extend this approach to support the documentation and development of trustworthy ICT systems, as depicted in Fig. 2. In our approach, we will re-use existing meta models for security standards, e.g., Sunyaev [12] and for risk management standards, e.g., Fenz [13] and combine them into a pattern for security and risk management standards (1). As a next step we will develop relations from these patterns to the CF (2), which allows us to re-use the existing relations to SRE methods (3). We combine the relations 1, 2, and 3 and, thus, we can create transitive relations the SRE methods to multiple security and risk management standards, e.g. ISO 27001 and Common Criteria (4).

However, the privacy and compliance demands of trustworthy ICT systems and security standards, e.g., ISO 27001 and Common Criteria, alike are not yet addressed. Hence, we propose to develop relations between specific patterns for laws (5), risk and security (6), and privacy (7). We will also extend the CF to enable relations to privacy and law extensions of SRE methods. The risk and security patterns shall address issues that are not already covered by an existing SRE method in 3. We will also develop the patterns in 5, 6, and 7, if there are no suitable patterns available yet. As a last step we combine the relations 5, 6, and 7 and, thus, also relate the patterns to multiple security standards, e.g. ISO 27001 and Common Criteria (8).

We choose cloud computing as an example of our work. Hence, we will create more detailed patterns for cloud systems based upon the aforementioned Cloud System Analysis Pattern.

Moreover, aligning clouds to meet compliance regulations is a challenging task, because of a high number of different kinds of stakeholders. We will address this problem by creating specific cloud law analysis patterns as an extension to our existing law pattern approach [9]. Our extension will also make use of results generated by the application of the cloud system analysis pattern.

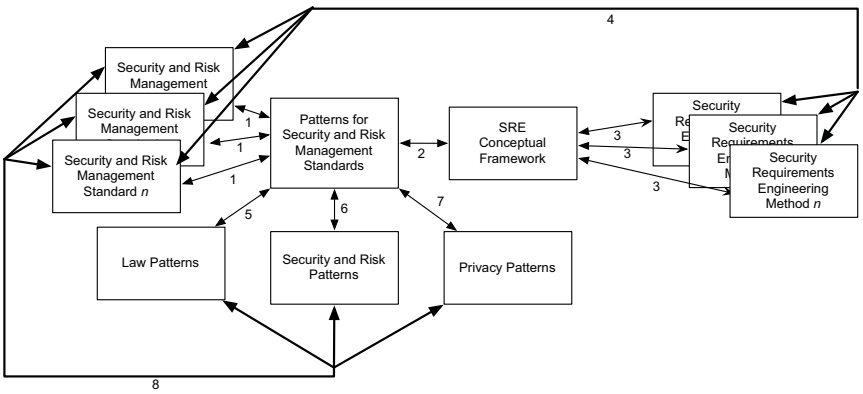

Fig. 2: Support for Developing and Documenting Trustworthy ICT Systems

We will start working on privacy patterns based upon Nissenbaum's model of informational privacy in terms of contextual privacy [14]. The model considers the context of a given situation, the kind of information and the relation of the information to the context. We will also compare security and risk management patterns using existing surveys, e.g., Heyman et al. [15].

The outcome of our work is a methodology for developing and documenting ICT systems with the goal to be compliant to security standards. We aim at developing a system of patterns supported by security requirements engineering approaches, which can be used to improve the security of an ICT system, as well as to generate a documentation of an ICT system. This documentation can be used as a basis for certification according to a standard.

The patterns in our work will be based upon UML and the problem frame approach by Michael Jackson [16]. In addition, essential parts of the patterns are specified with a formal notation based upon the Z notation [17]. The patterns will be derived from relevant scientific literature, existing pattern libraries, as well as being found in existing implementations of security standards.

We plan to validate our work via using the methodology and the pattern system for an ICT system and a specific security standard. We will compare the resulting documentation against a standard-compliant documentation that is not based on our patterns.

We conclude with a brief summary of the main benefits of our approach:

- A methodology for systematic pattern-based development and documentation of ICT systems
- Complementing patterns with existing SRE approaches in order to completely support the implementation of sections of security standards
- Specific-patterns for laws, privacy, security and risk management to cover all quality requirements of security standards
- Ease the burden of implementing security standards

References

1. Pfleeger, C.P., Pfleeger, S.L.: Security In Computing. 4th edn. Prentice Hall PTR (2007)
2. Anderson, R.: Security EngineerIng. 2nd edn. Wiley (2008)
3. ISO/IEC: Information technology - Security techniques - Information security management systems - Requirements. ISO/IEC 27001, International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) (2005)
4. Gamma, E., Helm, R., Johnson, R., Vlissides, J.M.: Design Patterns: Elements of Reusable Object-Oriented Software. 1 edn. Addison-Wesley Professional (1994)
5. Bishop, M.: Computer Security : art and science. 1st edn. Pearson (2003)
6. Beckers, K., Jürjens, J.: Security and compliance in clouds. In: Information Security Solutions Europe (ISSE 2010). Securing electronic business processes : Highlights of the Information Security Solutions Europe, Vieweg + Teubner (2010) 91–100
7. ISO/IEC: Information technology - security techniques - information security risk management. ISO/IEC 27005, International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) (2008)
8. Beckers, K., Küster, J.C., Faßbender, S., Schmidt, H.: Pattern-based support for context establishment and asset identification of the ISO 27000 in the field of cloud computing. In: Proceedings of the International Conference on Availability, Reliability and Security (ARES), IEEE Computer Society (2011) 327–333
9. Beckers, K., Küster, J.C., Faßbender, S., Schmidt, H.: A pattern-based method for identifying and analysing laws. In: REFSQ. (2012) to be published.
10. Beckers, K., Faßbender, S., Heisel, M., Küster, J.C., Schmidt, H.: Supporting the development and documentation of ISO 27001 information security management systems through security requirements engineering approaches. In: Proceedings of the International Symposium on Engineering Secure Software and Systems (ESSoS). LNCS, Springer (2012) to be published.
11. Fabian, B., Gürses, S., Heisel, M., Santen, T., Schmidt, H.: A comparison of security requirements engineering methods. Requirements Engineering – Special Issue on Security Requirements Engineering **15**(1) (2010) 7–40
12. Sunyaev, A.: Health-Care Telematics in Germany: Design and Application of a Security Analysis Method. 1st edn. Gabler Verlag (2011)
13. Fenz, S., Ekelhart, A., Neubauer, T.: Information security risk management: In which security solutions is it worth investing? Communications of the Association for Information Systems **28**(1) (5 2011) 329–356
14. Nissenbaum, H.: Privacy in Context: Technology, Policy, and the Integrity of Social Life. 1st edn. Stanford (2009)
15. Heyman, T., Scandariato, R., Huygens, C., Joosen, W.: Using security patterns to combine security metrics. In: Proceedings of the International Conference on Availability, Reliability and Security (AReS), IEEE Computer Society (2008) 1156–1163
16. Jackson, M.: Problem Frames. Analyzing and structuring software development problems. Addison-Wesley (2001)
17. ISO/IEC: Information technology – Z formal specification notation – Syntax, type system and semantics. ISO/IEC 13568, International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) (2002)

The dark side of vulnerability exploitation: a proposal for a research analysis. *

Ph.D. Student: Luca Allodi
Advisor: Prof. Fabio Massacci

Università degli studi di Trento
Trento, Italy

Abstract. Software security research has put much effort in evaluating security as a function of the expected number of vulnerabilities and their criticality. As hackers become more sophisticated and economically-driven, I argue that exploitation activities are a much more interesting index of risk than the number of vulnerabilities: the economics of the black market can shed light on attacking processes and trends, and can be very useful in better assessing security and re-thinking patching behavior and patches priority.

1 The problem is in the approach

Security is not easy to quantify. The usual approach [1–4] is to evaluate security in a semi-static way: the researcher takes into account the number of vulnerabilities that affect a system in some time-frame and their respective exploitation easiness; the vendor has to choose which vulnerabilities' patches should be prioritized and typically uses the CVSS Impact Score [5] as an index to make that decision. The main claim of my research proposal is that both the researcher and the vendor should not only be concerned with the volume and criticality of vulnerabilities, but rather with the *effective risk factor* that those would introduce in the operational system. I argue indeed that a vulnerability represents a risk only if it is not latent and is efficiently exploited.

I show in Section 3 that, while on first look this could seem an optimistic and naive claim, and actually an '*under*-semplification' of the problem, preliminary evidence exists that attackers' intentions are more predictable than considered in the literature: precious information can be inferred by more carefully monitoring criminal activities and exploitation trends. This type of information might seem hardly reliable and trustworthy, but the fact that criminal underground activity is becoming, as I show in section 3, more and more structured and economically-driven makes it easier and more meaningful to evaluate risk as related to actual criminal behavior and trends rather than simply as an unproven 'exposure to potential attacks'. My Ph.D. research goal is to identify, by means of auditing and understanding criminal underground activities, schemes or trends that would help in better defining security metrics and would be useful for: (a) the user,

* Work partially supported by EU-funded project FP7- 256980 NESSOS.

that could better choose which software configuration is more secure in that particular time-frame; (b) the vendor, that could better allocate human and economic resources by means of a more knowledgeable understanding of actual exploitation risks, thus increasing product security and monetization of effort.

In the next section I give some brief definitions helpful in describing vulnerability exploitation. In section 3 I introduce the different markets involved in the process and describe the structure supporting it. Finally in section 4 I draw my conclusions and Ph.D. proposals, formulating three hypothesis that, if hold true, could help improve software security and patches scheduling.

2 Some quick definitions

Vulnerabilities. I use Ozment's definition of vulnerability [6], according to which vulnerabilities are mistakes in the code or in the configuration of a software that can cause violations in its security policy. These mistakes can be exploited by an attacker to get access to the vulnerable system.

Exploits. One could identify different levels of maturity of an exploit as they usually are born as simple proofs-of-concept, are then scripted and eventually automated [7]. Frei et al. in [8] analyzed more than 14 thousand vulnerabilities. Out of these only about 3400 were exploited, most of which within a month from the disclosure of the vulnerability.

Attacks. An attacker needs to exploit (at least) one vulnerability in the system to reach his goal. The relation between exploitation time and vulnerability disclosure date is shown in [9] by Arora et. al: attacks increase at the time of the vulnerability disclosure. There also seems to be a correlation between random-wide information scans and attack probes, evidencing that untargeted attacks are common practice [10, 11].

3 The Markets

Vulnerability and exploit markets are distinct but related: while the former is divided between legitimate and illegitimate markets [12], the latter is mostly an underground activity usually labeled as 'black market'. On the other hand, the financial consequences of vulnerability exploitation have been shown to go far from solely their market value [13–15].

The market of vulnerabilities. No extended study exists, to the best of my knowledge, on the value of vulnerabilities in the black market. In [12] a very interesting insight on the legitimate vulnerability market is given; there are many difficulties in the legitimate selling of vulnerabilities to vendors, because of the 'secretive' nature of the good. The relationship between the software vendor and the security researcher, especially if independent and external to the company, can be trouble[1]: the vendor may indeed not appreciate the bad publicity that the disclosure of a vulnerability earns him [16].

[1] http://news.cnet.com/8301-27076_3-57320190-248/apple- boots- security-guru-who-exposed-iphone-exploit/

The market of attacks. The economical value for the attacker seems significant: in [13] Franklin et al. investigate the amount of financial and economical-related information that circulate in the market; they calculate the value of the market *of the credit cards only* to be about 37million USD; if one considers bank data theft and identity theft, their estimation increases up to 93million USD. The magnitude of these estimates is also confirmed in [17]. Motoyama et al. in [14] study the dynamics of underground forums, in which these data are actually traded, and show the high interest the criminals put in online payment accounts and stolen financial information.

The market of exploits. Lately spam has become a way to diffuse malicious links that drive the user toward domains controlled by the attackers, that can then try (and often succeed) to exploit their systems; very diffuse vectors for such an activity are porn sites [18], botnets [15, 19, 20], and social networks [21]. Once the attacker gets access to the victim's machine, he can install keyloggers or any kind of malware that will provide him with the victim's private data or 'permanent' access to the machine, at his will.

The profile of the coders that write exploits may vary a lot, ranging from security enthusiasts to professionals. Some coders put a lot of effort in efficiently exploiting vulnerabilities; these 'efficient' exploits are featured in web applications with a MySql backend; the community calls them *Exploit Kits*.

It is my opinion that the Exploit Kits phenomenon can shed some light on the exploiting economics underlying the whole data-theft market, and due to some of its peculiarities can perhaps be of great value in better evaluating *effective risk*. Moreover, it provides preliminary evidence that exploiting activities are governed by an economical process not yet investigated by the scientific community.

There are many different Exploit Kits on the market, very often advertised in underground forums such as *exploit.in* and *vendors.pro*. Examples of these are Phoenix, Eleonore, Blackhole, Crimepack. Exploit kits are rented to the interested attacker for different periods of time, usually up to an year; a one-year license would cost from 1000USD to 2500USD[2]. Perhaps the most popular Exploit Kit around is now Blackhole[3], but Phoenix and others have a significant market share too. Their coders seem to put a lot of effort in code obfuscation and encryption[4]. Even more importantly and perhaps counter-intuitively, but supporting the hypothesis that exploitation is driven by economical processes, the number of exploited vulnerabilities in these packs is in the order of ten or less, and many of them are very old ones.

As an example, these are the softwares exploited in Eleonore v1.6.5, released in March 2011[5] featuring only 10 exploits, most of which are at least 1-year old and two of which are 5+ years old: MDAC(2006), WMI Object Broke (2006),

[2] http://malwareint.blogspot.com/2010/01/state-of-art-in-eleonore-exploit-pack.html
[3] http://dvlabs.tippingpoint.com/blog/2011/04/26/blackhole-exploit-kit
[4] http://research.zscaler.com/2011/02/blackhole-exploits-kit-attack-growing.html
[5] http://exploit.in/forum/index.php?showtopic=46653 (account required to access the page; the reader might want to use a TOR network or a secure proxy to access the page, depending to whom belongs the IP used)

Snapshot (2008), IEpeers (2010), HCP (2010), PDF libtiff mod v1.0 (2010), Flash <10.2 (2011), Flash < 10.2.159 (2011), Java Invoke (2010), Java trust (2010). Analogous are Blackhole's[6] and Phoenix's[7] offerings, as many others' too[8]. The vulnerabilities in those Exploit Kits concern a small set of widely diffused and exposed softwares such as Java, Flash, or Adobe Reader plugins; while at the time of writing Java seems to be the main target in the most diffused exploit kits[9] (Blackhole, Phoenix), in the past were mainly targeted Office Plugins and Flash[10], suggesting there might be additional, software-related trends in the process. Exploit kits are advertised by screenshots and exploiting success rates[11].

The actual exploitation takes place when the victim requests the, say, 'exploit.php' page on the attacker's domain[12]. The attacker must fool the user in requesting that web-page: apart from social engineering and direct link spam techniques, the attacker usually compromises one or more hosts (often by means of SQL Injection) and insert an iframe in the domain's homepage that redirects connections towards the attacker's 'exploit.php' page; this is a very common practice, as evidenced by sites such as Malware Domain List[13] that serve as a database of hosts that have been compromised. Once the victim reaches the attacker's host, a set of exploiting scripts is run; as a consequence, the successful attacker can often execute code on the target machine: install keyloggers, steal data, download malware and/or make the machine part of his botnet. In order to increase the hit rate, the compromised sites might be acquired by somebody else; the attacker could (and this may not be an exhaustive list):

- buy a set of hosts compromised by somebody else
- rent connections to compromised hosts from whom acquired them
- rent connections from traffic brokers[14,15] that buy traffic from some third party (2-6USD per 1k connections).

In particular, the second approach is made easier by the existence of traffic dispatchers (e.g. SimpleTDS[16]), and often augmented by botherders themselves [22]; the third is widely diffused in pay-per-click(-install) scenarios such as porn networks [18] and others [22]: the traffic from a compromised host is sold by the 'compromiser' to the traffic broker, which will then receive a certain amount of connections from victims that accessed the compromised host. These

[6] http://exploit.in/forum/index.php?showtopic=41662
[7] http://exploit.in/forum/index.php?showtopic=37627
[8] http://vil.nai.com/images/FP_BLOG_100527_1.jpg
[9] http://www.kaspersky.com/about/news/virus/2011/Ja va_the_Target_of_ Choice_for_Exploit_Kits_in_2011
[10] https://threatpost.com/en_us/blogs/carberp-and-black-hole-exploit- kit-wreaking-havoc-120511
[11] http://malwareint.blogspot.com/2010/09/black-hole-exploits-kit-another.html
[12] http://blog.imperva.com/2011/12/deconstructing-the-black-hole- exploit-kit.html
[13] http://www.malwaredomainlist.com/
[14] http://www.trafficshop.com/
[15] http://www.trafficholder.com/
[16] http://www.simpletds.com/

very connections are redirected by the traffic broker to his clients, that in turn have bought a certain amount of 'traffic' that will be directed straight to the 'exploit.php' page under their control [18, 22].

4 A research plan

From this preliminary analysis, the exploitation market results far from being simply driven by enthusiasts, unorganized hackers or groups of hackers: there is a whole infrastructure supporting both the exploitation of vulnerabilities and the economic investment that the attacker must (and apparently actually do) sustain. This gives preliminary evidence and, to my opinion, a very good reason to further investigate the dynamics of exploitation and the attackers' goals, in order to provide insights about actual security and perhaps, eventually, to better evaluate security metrics, countermeasures, risk assessment and to support vendors' patching behavior.

My research goal is to find a novel, more precise way to describe vulnerability exploitation, and thus to evaluate the *effective risk factor* affecting a system. In order to accomplish that, I formulate the following three hypotheses:

- **Hypothesis (1).** Attackers are economically rational.
- **Hypothesis (2).** There is a substantial difference in success rates between public and commercial exploits.
- **Hypothesis (3).** Commercial exploits are not redundant (i.e. not many exploits exist in the same time-frame for the same system configuration).

Therefore by (2) higher risk would come from those vulnerabilities for which a commercial exploit exists; if (1) holds, then the most dangerous vulnerabilities will be those that are efficiently exploitable, because those would optimize the exploitation success rate and thus maximize attackers' return on investment. Following (3) vulnerabilities that provide access to a certain system configuration for which other, easier or more efficiently exploitable vulnerabilities exist would represent a lower risk because of less interest to the attacker.

I'm planning to investigate those hypothesis during my Ph.D. program here at the University of Trento. **Hypothesis 3** can be validated by analyzing hackers' exploitation resources; I'm planning to further understand how much diffused those tools are as attack vectors. I'm also willing to understand who is behind their development and how profitable this activity is. **Hypothesis 2** will involve testing the efficacy of publicly released exploits against the ones featured in exploitation tools from (3). *Dulcis in fundo*, **Hypothesis 1** will be the toughest one to investigate: to collect evidence of the importance of the economic aspects in the attacking process may not be sufficient; I'm planning to conduct interviews with (professional) hackers and to design and deploy a social experiment with the purpose of better understanding how much effort the attackers are willing to put into the exploitation of a system.

The validity of those hypotheses could smooth the way toward a more precise and realistic risk assessment process, and significantly improve security metrics's reliability, patching priorities, and system hardening efficiency and efficacy.

References

1. M. Howard, J. Pincus, and J. Wing, "Measuring relative attack surfaces," *Comp. Sec. in the 21st Century*, pp. 109–137, 2005.
2. P. K. Manadhata and J. M. Wing, "An attack surface metric," *TSE*, vol. 37, pp. 371–386, 2011.
3. I. Kotenko and M. Stepashkin, "Attack graph based evaluation of network security," in *Proc. of CMS'06*, ser. LNCS. Springer, 2006, vol. 4237, pp. 216–227.
4. L. Wang, A. Singhal, and S. Jajodia, "Measuring the overall security of network configurations using attack graphs," in *Proc. of DAS'07*, 2007, pp. 98–112.
5. P. Mell and K. Scarfone, *A Complete Guide to the Common Vulnerability Scoring System Version 2.0*. CMU, 2007.
6. A. Ozment, "Improving vulnerability discovery models," in *Proc. of QoP'07*, ser. QoP '07. New York, NY, USA: ACM, 2007, pp. 6–11.
7. W. Arbaugh, W. Fithen, and J. McHugh, "Windows of vulnerability: a case study analysis," *Computer*, vol. 33, no. 12, pp. 52 – 59, 2000.
8. S. Frei, M. May, U. Fiedler, and B. Plattner, "Large-scale vulnerability analysis," in *Proc. of LSAD'06*. ACM, 2006, pp. 131–138.
9. A. Arora, R. Krishnan, A. Nandkumar, R. Telang, and Y. Yang, "Impact of vulnerability disclosure and patch availability-an empirical analysis," in *Proc. of WEIS'04*, 2004.
10. S. Ransbotham and S. Mitra, "Choice and chance: A conceptual model of paths to information security compromise," *ISR*, vol. 20, 2009.
11. B. W., H. M., H. A., and H. C.David, "2011 data breach investigation report," Verizon, Tech. Rep., 2011.
12. C. Miller, "The legitimate vulnerability market: Inside the secretive world of 0-day exploit sales," in *Proc. of WEIS'07*, 2007.
13. J. Franklin, V. Paxson, A. Perrig, and S. Savage, "An inquiry into the nature and causes of the wealth of internet miscreants," in *Proc. of CCS'07*, ser. CCS '07, 2007, pp. 375–388.
14. M. Motoyama, D. McCoy, S. Savage, and G. M. Voelker, "An analysis of underground forums," in *Proc. of IMC'11*, 2011.
15. C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage, "Spamalytics: an empirical analysis of spam marketing conversion," in *Proc. of CCS'08*, ser. CCS '08. ACM, 2008, pp. 3–14.
16. R. Anderson and T. Moore, "The economics of information security," *Science*, vol. 314, p. 610, 2006.
17. A. Cárdenas, S. Radosavac, J. Grossklags, J. Chuang, and C. Hoofnagle, "An economic map of cybercrime," in *Proc. of TPRC'09*, 2009.
18. G. Wondracek, T. Holz, C. Platzer, E. Kirda, and C. Kruegel, "Is the internet for porn? an insight into the online adult industry," in *Proc. of WEIS'10*, 2010.
19. B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: analysis of a botnet takeover," in *Proc. of CCS'09*. ACM, 2009.
20. J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon, "Peer-to-peer botnets: overview and case study," in *Proc. of HOTBOTS'07*, 2007.
21. T. Kurt, G. Chris, P. Vern, and S. Dawn, "Suspended accounts in retrospect:an analysis of twitter spam," in *Proc. of IMC'11*. ACM, 2011.
22. J. Baltazar, "More traffic, more money: Koobface draws more blood," TrendLabs, Tech. Rep., 2011.